AN ε-OPTIMAL SEQUENTIAL APPROACH FOR SOLVING ZS-POSGS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent reductions of finite-horizon two-player zero-sum partially observable stochastic games (zs-POSGs) admit dynamic programming with linear programming backups, but these remain computationally demanding. Building on sequential central planning for cooperative games, we factorise the optimisation over the joint decision rule into two planning sub-stages per game stage, one per player. This ordering preserves the game value and the players' information constraints while their actions remain simultaneous. We establish the sufficiency of sequential occupancy states for the value recursion and show that both sub-stages retain the maximum-of-concave structure of the simultaneous formulation. We integrate the resulting linear programming backups into a sequential point-based value iteration algorithm ($\texttt{PBVI}_{\texttt{seq}}$). We bound the exploitability of its policies, accounting for approximation at both sub-stages. On several medium-to-large benchmarks, $\texttt{PBVI}_{\texttt{seq}}$ returns policies with lower exploitability than competing solvers and solves horizons at which some baselines exceed time or memory limits.

1 INTRODUCTION

The principle of optimality, introduced by Bellman, underpins dynamic programming for fully observable Markov Decision Processes (MDPs) (Bellman, 1957) and zero-sum Stochastic Games (SGs) (Shapley, 1953). In simultaneous-move two-player zero-sum Partially Observable Stochastic Games (zs-POSGs), however, each player selects actions using only its own action–observation history. Planning must therefore satisfy these information constraints while accounting for uncertainty about the state and the opponent's private history. Since the number of possible private histories can grow exponentially with the horizon, computing policies with low exploitability can be costly (Hansen et al., 2004).

Significant progress has been made in the adjacent field of cooperative POSGs (Dec-POMDPs). From a centralised planning perspective, these problems can be reformulated as continuous-state MDPs (Nayyar et al., 2013; Oliehoek, 2013; Dibangoye et al., 2013). This approach facilitated the transfer of dynamic programming techniques (Dibangoye et al., 2016; Dibangoye & Buffet, 2018). Together with advances in heuristic search (Szer et al., 2005; Oliehoek et al., 2010; Dibangoye et al., 2012; Peralez et al., 2024), these developments have expanded the range of cooperative problems that can be solved. In zs-POSGs, however, the minimax structure complicates both value computation and policy extraction. Approaches relying on heuristic search (Horák et al., 2017; Horák & Bošanskỳ, 2019; Buffet et al., 2020a;b), search combined with deep learning (Brown et al., 2020; Moravcík et al., 2017), or regularised public-belief games (Sokota et al., 2023) offer different solutions. Within dynamic programming, structural properties of value functions (Wiggers et al., 2016; Cunha et al., 2023; Nayyar & Gupta, 2017) have supported further progress. In particular, Delage et al. (2023) maintain the strategies associated with their value bounds to recover executable policies.

More recently, Escudie et al. (2025) show that finite-horizon two-player zs-POSGs can be losslessly reduced to transition-independent zero-sum stochastic games (TI-zs-SGs). This reduction preserves the game value and information constraints while supporting dynamic programming over a structured planning model. In this formulation, the planner reasons over *occupancy states*—distributions over hidden states and joint histories—and selects decision rules, which map private histories to action distributions. The authors also establish a maximum-of-concave representation of the optimal value function and use it to formulate greedy updates as linear programmes. However, polynomial size in

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

the explicitly stored value representation does not ensure a low computational cost: the representation itself can grow with the horizon. Moreover, the backup handles the players' decision rules through a simultaneous minimax formulation. As the number of private histories and the stored representation grow, the resulting programmes can become expensive to solve.

To address this computational cost, we factorise the optimisation over the joint decision rule into two planning sub-stages per game stage, one per player. This ordering concerns the planner's computations: during execution, the players still act simultaneously and use only their own action–observation histories. Our construction builds on the sequential central-planning framework of Peralez et al. (2025), who introduced sequential occupancy states for Dec-POMDPs and established a piecewise-linear convex value structure. Extending this construction to zs-POSGs requires preserving the minimax value, together with the information constraints and value-function structure needed for the sequential updates.

Our contributions are threefold. First, we establish that sequential occupancy states are sufficient for the value recursion at each planning sub-stage in zs-POSGs. We prove that the reformulation preserves the game value and the players' information constraints. We also show that the maximum-of-concave structure established for the simultaneous formulation is preserved at both sub-stages. Second, we use this structure to derive separate linear programming formulations of the backup. We integrate these backups into point-based value iteration (PBVI) (Pineau et al., 2003) and call the resulting algorithm sequential PBVI ($\texttt{PBVI}_{\texttt{seq}}$). For this algorithm, we bound policy exploitability while accounting for approximation at both planning sub-stages. Third, we evaluate $\texttt{PBVI}_{\texttt{seq}}$ on challenging benchmarks. On several medium-to-large benchmarks, $\texttt{PBVI}_{\texttt{seq}}$ returns policies with lower exploitability than competing solvers and solves horizons at which some baselines exceed time or memory limits.

2 Preliminaries

In this section, we formally define zero-sum partially observable stochastic games (zs-POSGs) and establish the notation used throughout our work. We then recall the occupancy-based planning formulation and its linear programming backups.

2.1 Simultaneous-Move zs-POSGs

A simultaneous-move zs-POSG is defined by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}_1, \mathcal{A}_2, \mathcal{Z}_1, \mathcal{Z}_2, p, r, b, \gamma, \ell)$. The game proceeds over a finite horizon ℓ, involving two competing agents, player 1 (maximiser) and player 2 (minimiser). The finite set $\mathcal{S}$ represents the hidden states of the environment. The action sets $\mathcal{A}_i$ and observation sets $\mathcal{Z}_i$ are also finite. At each stage $t \in \{0, \dots, \ell - 1\}$, both players simultaneously select private actions $a_1 \in \mathcal{A}_1$ and $a_2 \in \mathcal{A}_2$. These actions, combined with the current state s, trigger a stochastic transition to a next state s' and the generation of observations according to the joint probability kernel $p(s', z_1, z_2 \mid s, a_1, a_2)$. Here, $z_i \in \mathcal{Z}_i$ denotes the private observation received by player i. The observations may include a common component observed by both players.[1] Player 1 receives the immediate payoff $r(s, a_1, a_2)$, and player 2 receives its negative. The game starts from a distribution $b \in \Delta(\mathcal{S})$ and discounts future rewards by $\gamma \in [0, 1)$.

Unlike in fully observable games, players act here under imperfect information. Before selecting its action at stage t, player i has access to its private history, $h_{i,t} = (a_{i,0}, z_{i,1}, \dots, a_{i,t-1}, z_{i,t})$, and $h_{i,0} = \varnothing$. We denote the set of these histories by $\mathcal{H}_{i,t}$. To handle this uncertainty, players use behavioural *policies* $\pi_i = (d_{i,0}, \dots, d_{i,\ell-1})$, i.e., sequences of history-dependent decision rules. Each rule $d_{i,t} : \mathcal{H}_{i,t} \to \Delta(\mathcal{A}_i)$ maps private histories to action distributions. We denote by $\mathcal{D}_{i,t}$ the set of such rules at stage t. The value of the game under a joint policy (π_1, π_2) is the expected discounted return from the initial belief b, given by $v_{\pi_1,\pi_2}(b) = \mathbb{E}_{b,\pi_1,\pi_2}[\sum_{t=0}^{\ell-1} \gamma^t r(s_t, a_{1,t}, a_{2,t})]$. Since each player remembers its own actions and observations, the game has perfect recall. By Kuhn's theorem (Kuhn, 1953), behavioural policies are therefore realisation-equivalent to mixed strategies over deterministic policies. Applying the minimax theorem to this finite normal-form representation (Neumann, 1928; Sion, 1958) gives the unique game value $v^*_{\text{sim}}(b) = \max_{\pi_1} \min_{\pi_2} v_{\pi_1,\pi_2}(b) = \min_{\pi_2} \max_{\pi_1} v_{\pi_1,\pi_2}(b)$. Our goal is to compute an ε-optimal policy π_1 for the first player, satisfying

[1] For readability, we omit separate notation for public observations. The sequential formulation accommodates them through the same conditioning on public histories as in Escudie et al. (2025).

$v^*_{\text{sim}}(b) - \min_{\pi_2} v_{\pi_1,\pi_2}(b) \leq \varepsilon$. We refer to the difference on the left-hand side as the exploitability of π_1.

2.2 Occupancy-Based Planning

Extending Bellman's principle of optimality to finite-horizon γ-discounted zs-POSGs requires an information state for planning. Because neither player generally observes the state or the joint history, the resulting formulation must preserve the information constraints on their policies. We call a reduction *lossless* when it preserves policy values, equilibria and the players' information constraints (Sanjari et al., 2023). In particular, the correspondence between admissible policies must preserve expected returns and unilateral deviations in both directions.

Existing approaches construct planning states from different sources of information. Public-belief formulations use shared information (Ghosh et al., 2004; Horák & Bošanskỳ, 2019), whereas one-sided formulations exploit one player's knowledge of the state (Horák et al., 2017). Public-belief planning supports both ReBeL (Brown et al., 2020) and public-belief announcement games (Sokota et al., 2023). For the latter, suitable regularisation yields equilibrium correspondence and bounds on exploitability. Occupancy Markov games (zs-OMGs) use *occupancy states* for centralised planning (Wiggers et al., 2016; Buffet et al., 2020a; Delage et al., 2023). An occupancy state is the distribution over the current state and joint histories induced by b and the joint decision rules used before stage t. Although these distributions support Bellman-style value recursions, their use for planning does not make them observable to the players. Policy extraction must therefore respect the original private histories. To this end, Delage et al. (2023) maintain the strategies associated with their value bounds and recover executable policies with approximation guarantees.

Building on occupancy-based planning, Escudie et al. (2025) provide a lossless reduction from finite-horizon zs-POSGs to transition-independent zero-sum stochastic games. Their formulation distinguishes the occupancy state used for value computation from player-specific occupancy sets used for policy extraction. Each such set contains the occupancies consistent with a player's own past decision rules and the possible past decision rules of its opponent. Policy extraction thus does not require either player to observe the global occupancy. For value computation, they establish a maximum-of-concave representation of the optimal value function. Their point-based value iteration ($\texttt{PBVI}_{\texttt{sim}}$) method exploits this structure by storing a finite collection of concave, piecewise-linear functions, called envelopes, whose pointwise maximum defines the value approximation. This representation allows greedy updates to be formulated as linear programmes without enumerating all deterministic opponent decision rules. The size of these programmes is polynomial in the explicit stored representation. However, the resulting programmes can still be expensive to solve. Their variables and constraints depend on the stored envelopes and the opponent's histories, actions and observations. The representation can grow across $\texttt{PBVI}_{\texttt{sim}}$ iterations, while the number of private histories can grow exponentially with the horizon. To address this computational cost, the next section develops a sequential factorisation of the optimisation over the joint decision rule.

3 The Sequential Reformulation

The simultaneous backup performs a minimax optimisation over the joint space of decision rules $\mathcal{D}_{1,t} \times \mathcal{D}_{2,t}$. We represent this optimisation through two planning sub-stages per game stage t, one for each player. The maximisation at $(1, t)$ uses the value of the minimisation at $(2, t)$. This ordering concerns the planner's computations. During execution, the players still act simultaneously, each using its own action–observation history.

3.1 Sequential Occupancy States

To represent these two planning sub-stages, we adapt the sequential occupancy construction developed for finite-horizon γ-discounted Dec-POMDPs to zs-POSGs (Peralez et al., 2025). The first occupancy, $x_{1,t}$, is the distribution at the beginning of game stage t. The second, $x_{2,t}$, extends this distribution to include player 1's action under the selected decision rule.

Definition 3.1. Let $\theta_{1,t} \doteq (d_{1,0}, d_{2,0}, \ldots, d_{1,t-1}, d_{2,t-1})$ and $\theta_{2,t} \doteq (\theta_{1,t}, d_{1,t})$ be the decision-rule prefixes at planning sub-stages $(1, t)$ and $(2, t)$, respectively, with $\theta_{1,0} = \varnothing$. For initial distribution b,

for any $(t, s, h_{1,t}, h_{2,t}, a_1)$, the sequential occupancy states are

$$x_{1,t}(s, h_{1,t}, h_{2,t}) \doteq \Pr(s, h_{1,t}, h_{2,t} \mid b, \theta_{1,t}), \quad x_{2,t}(s, h_{1,t}, h_{2,t}, a_1) \doteq \Pr(s, h_{1,t}, h_{2,t}, a_1 \mid b, \theta_{2,t}).$$

The internal update induced by $d_{1,t}$ is $x_{2,t}(s, h_{1,t}, h_{2,t}, a_1) = x_{1,t}(s, h_{1,t}, h_{2,t})\, d_{1,t}(a_1 \mid h_{1,t})$.

Figure 3 illustrates the resulting sequence $x_{1,t} \xrightarrow{d_{1,t}} x_{2,t} \xrightarrow{d_{2,t}} x_{1,t+1}$. The sequential occupancy records a distribution over player 1's action and is internal to planning. Player 2 receives no additional information from the sequentialisation before choosing its action at game stage t. Its executable decision rule retains the domain $d_{2,t} : \mathcal{H}_{2,t} \to \Delta(\mathcal{A}_2)$. We write $\mathcal{X}_{\text{seq}}$ for the set of occupancies induced from b by admissible decision-rule prefixes. Each state includes its planning sub-stage (i, t): states $x_{1,t}$ occur at $t = 0, \ldots, \ell$, and states $x_{2,t}$ at $t = 0, \ldots, \ell - 1$.

3.2 Sequential Dynamics and Rewards

We denote the sequential transition and reward by τ_{seq} and ρ_{seq}. At the first planning sub-stage, applying $d_{1,t}$ extends the distribution without advancing the environment. We therefore set $\tau_{\text{seq}}(x_{1,t}, d_{1,t}) \doteq x_{2,t}$, using the update in Definition 3.1, and $\rho_{\text{seq}}(x_{1,t}, d_{1,t}) \doteq 0$. At the second planning sub-stage, applying $d_{2,t}$ completes the joint action distribution. The expected reward is

$$\rho_{\text{seq}}(x_{2,t}, d_{2,t}) \doteq \sum_{s, h_{1,t}, h_{2,t}, a_1, a_2} x_{2,t}(s, h_{1,t}, h_{2,t}, a_1)\, d_{2,t}(a_2 \mid h_{2,t})\, r(s, a_1, a_2).$$

The transition $\tau_{\text{seq}}(x_{2,t}, d_{2,t}) \doteq x_{1,t+1}$ uses the original environment kernel. For history extensions $h_{i,t+1} = (h_{i,t}, a_i, z_i)$, it is given by

$$x_{1,t+1}(s', h_{1,t+1}, h_{2,t+1}) = \sum_s x_{2,t}(s, h_{1,t}, h_{2,t}, a_1)\, d_{2,t}(a_2 \mid h_{2,t})\, p(s', z_1, z_2 \mid s, a_1, a_2).$$

Discounting applies only when advancing to the next game stage. We therefore use $\gamma_i = \gamma^{(i-1)}$.

Definition 3.2. The sequential occupancy planning model is $\mathcal{M}_{\text{seq}} \doteq (\mathcal{X}_{\text{seq}}, \mathcal{F}_{\text{seq}}, \mathcal{D}, \tau_{\text{seq}}, \rho_{\text{seq}}, \gamma, \ell)$, where $\mathcal{D} \doteq \cup_{t=0}^{\ell-1}(\mathcal{D}_{1,t} \cup \mathcal{D}_{2,t})$. At each nonterminal planning sub-stage (i, t), the admissible rules are precisely those in $\mathcal{D}_{i,t}$. The initial occupancy satisfies $x_{1,0}(s, \varnothing, \varnothing) = b(s)$. The terminal set $\mathcal{F}_{\text{seq}}$ consists of states at $(1, \ell)$, with zero terminal payoff. The horizon comprises ℓ game stages and 2ℓ planning sub-stages, with discount factors γ_1 and γ_2 as defined above.

3.3 Sequential Value Recursion

The sequential planner evaluates the max–min expected discounted return over the remaining game stages. Writing $d_{i,t:} = (d_{i,t}, \ldots, d_{i,\ell-1})$ for a sequence of remaining admissible decision rules, we define $v^*_{\text{seq}}(x_{i,t}) \doteq \max_{d_{1,t+i-1:}} \min_{d_{2,t:}} \sum_{k=t}^{\ell-1} \gamma^{k-t} \rho_{\text{seq}}(x_{2,k}, d_{2,k})$, where the occupancies follow τ_{seq} from $x_{i,t}$ under these rules. At planning sub-stage $(2, t)$, player 1's remaining sequence starts at $t + 1$, since its current rule is already incorporated into $x_{2,t}$. A sequence starting at ℓ is empty. The following proposition expresses this objective recursively, optimising one decision rule at a time.

Proposition 3.3 (Proof in App. A.2)**.** *For every terminal sequential occupancy state, $v^*_{\text{seq}}(x_{1,\ell}) = 0$. At every nonterminal planning sub-stage (i, t), for every sequential occupancy state $x_{i,t}$,*

$$v^*_{\text{seq}}(x_{i,t}) = \text{opt}_{i,\, d_{i,t} \in \mathcal{D}_{i,t}} \left[\rho_{\text{seq}}(x_{i,t}, d_{i,t}) + \gamma_i v^*_{\text{seq}}(\tau_{\text{seq}}(x_{i,t}, d_{i,t})) \right],$$

where $\text{opt}_1 = \max$ *and* $\text{opt}_2 = \min$.

The sequential reformulation preserves the simultaneous optimal value v^*_{sim} of the simultaneous occupancy formulation of Escudie et al. (2025) at every game-stage boundary.

Corollary 3.4 (Proof in App. A.2)**.** *For every sequential occupancy state $x_{1,t} \in \mathcal{X}_{\text{seq}}$, $v^*_{\text{seq}}(x_{1,t}) = v^*_{\text{sim}}(x_{1,t})$, for $t = 0, \ldots, \ell$. Identifying $x_{1,0}$ with b, we obtain $v^*_{\text{seq}}(b) = v^*_{\text{sim}}(b)$.*

The next section uses this recursion to derive the backup formulations. The correspondence with executable policies is established separately in Section 5.

4 SEQUENTIAL BACKUP GEOMETRY

The sequential recursion of Section 3 exposes an intermediate value at each game stage. This section characterises its geometry and uses it to factorise the simultaneous backup of Escudie et al. (2025) into two linear programmes (LPs), one per planning sub-stage. Both formulations are polynomial in the value function representation. The factorisation separates the current decision-rule variables from part of the continuation-value computation, reducing the dominant constraint-matrix coupling quantified in Theorem 4.4. Concretely, Section 4.1 derives the value representation from its supporting hyperplanes. Section 4.2 then derives the two greedy updates and compares the sizes of their LPs with that of the simultaneous-move backup. Section 4.3 integrates these updates into PBVI and establishes the approximation guarantee when no value representations are pruned.

4.1 SEQUENTIAL VALUE GEOMETRY

For a fixed suffix policy of player 1, player 2's best-response problem is a POMDP. Its value is therefore piecewise linear and concave in player 2's local belief and can be represented by a finite set of α-vectors (Pineau et al., 2003). The occupancy state can in turn be decomposed according to player 2's private histories, allowing this POMDP value to be evaluated on each resulting local weighted belief and the evaluations to be summed.

Decomposition into local weighted beliefs. For each private history h_2, we extract from the occupancy state the local weighted belief b_{h_2} on which player 2's best-response value is evaluated. This belief must capture the full latent state, which includes the opponent's private history h_1. Let $\Omega_1 \doteq \mathcal{S} \times \mathcal{H}_1$ and $\Omega_2 \doteq \mathcal{S} \times \mathcal{H}_1 \times \mathcal{A}_1$ denote the latent-state domains of player 2's best-response POMDP at sub-stages 1 and 2, respectively. For a given occupancy state x_i, define the local weighted belief $b_{h_2} \in \mathbb{R}_+^{\Omega_i}$ as the unnormalised slice $b_{h_2}(\omega) \doteq x_i(\omega, h_2)$, where $\omega \in \Omega_i$. The vector b_{h_2} contains both the conditional belief over the latent state and the probability mass of h_2. Hence evaluating an α-vector on b_{h_2} already accounts for the marginal probability of that private history, and summing the local values over h_2 recovers the value at the occupancy state.

From α-vectors to the value function. Fix a sub-stage $i \in \{1, 2\}$. For a fixed suffix policy of player 1, let $\Gamma_2 \subset \mathbb{R}^{\Omega_i}$ denote the finite set of α-vectors representing player 2's best-response POMDP. Its value at the local weighted belief associated with h_2 is $\min_{\alpha \in \Gamma_2} \langle b_{h_2}, \alpha \rangle$. The corresponding value at x_i is therefore the sum of these local values. Collecting sets Γ_2 in Γ_1 allows the optimisation over player 1's suffix policies to be written as the outer maximisation in Theorem 4.1.

Theorem 4.1 (Proof in App. B)**.** *Fix a sub-stage $i \in \{1, 2\}$. Let Γ_1 be the family of finite sets induced by player 1's admissible suffix policies, where each $\Gamma_2 \in \Gamma_1$ is a subset of $\mathbb{R}^{\Omega_i}$. For any occupancy x_i at this sub-stage, the optimal sequential value is*

$$v^*_{\text{seq}}(x_i) = \max_{\Gamma_2 \in \Gamma_1} \texttt{Val}_{\Gamma_2}(x_i), \qquad \textit{where} \quad \texttt{Val}_{\Gamma_2}(x_i) \doteq \textstyle\sum_{h_2} \min_{\alpha \in \Gamma_2} \langle b_{h_2}, \alpha \rangle.$$

Theorem 4.1 provides the value representation used to derive the greedy update at each sub-stage.

4.2 SEQUENTIAL GREEDY BACKUPS

We now use the above value representation to derive the greedy update at each planning sub-stage.

Optimising d_1. At sequential occupancy x_1, player 1 selects its current decision rule using the continuation value represented by Γ_1. Since the immediate reward at this sub-stage is zero, the greedy update maximises the value of the induced sequential occupancy.

Lemma 4.2 (Proof in App. C)**.** *Fix a finite family Γ_1 of sets Γ_2 representing the continuation values at the intermediate sub-stage. The greedy decision rule of player 1 at occupancy x_1 belongs to*

$$\arg\max_{d_1} \max_{\Gamma_2 \in \Gamma_1} \texttt{Val}_{\Gamma_2}\big(\tau_{\text{seq}}(x_1, d_1)\big).$$

Since Γ_1 is finite, we solve LP_1 for each $\Gamma_2 \in \Gamma_1$ and retain the pair (d_1^*, Γ_2^*) with the largest value.

$$\begin{aligned} &\text{maximise}_{d_1, w} \quad \textstyle\sum_{h_2} w_{h_2} \\ &\text{subject to} \quad w_{h_2} \le \langle b_{h_2}, \alpha \rangle, \quad \forall h_2,\ \forall \alpha \in \Gamma_2. \end{aligned} \tag{LP$_1$}$$

The decision variables d_1 enter LP$_1$ through the local weighted beliefs b_{h_2}. Indeed, $\langle b_{h_2}, \alpha \rangle = \sum_{s,h_1,a_1} x_1(s,h_1,h_2)\, d_1(a_1 \mid h_1)\, \alpha(s,h_1,a_1)$, which is linear in d_1. Hence LP$_1$ optimises the current decision rule for a fixed continuation representation Γ_2.

Optimising d_2. At sequential occupancy $x_{2,t}$, player 2 selects its current decision rule before the game proceeds to the next sub-stage $(1, t+1)$. Let Γ_1' represent the continuation value at that sub-stage. To evaluate these future values at the sequential occupancy, we back-project their α-vectors through the one-step transition. For $\Gamma_2' \in \Gamma_1'$, $\alpha \in \Gamma_2'$, a_2, and z_2, define the one-step back-projection

$$q^{\alpha}_{a_2,z_2}(s,h_1,a_1) \doteq r(s,a_1,a_2)/|\mathcal{Z}_2| + \gamma \textstyle\sum_{s',z_1} p(s',z_1,z_2 \mid s,a_1,a_2)\, \alpha(s', h_1 \cdot a_1 \cdot z_1).$$

The factor $1/|\mathcal{Z}_2|$ ensures that the current reward is counted once when summing over z_2. The greedy update must account for the possible continuations of player 1 represented by Γ_1'. Together with player 2's current decision, these continuations define the finite zero-sum continuation game whose LP formulation gives the greedy value at the sequential occupancy.

Lemma 4.3 (Proof in App. C)**.** *Fix a finite family Γ_1' representing the continuation value at the next sub-stage $(1, t+1)$. The greedy value at x_2 is given by equation LP$_2$.*

Together, LP$_1$ and LP$_2$ implement the greedy updates at the two planning sub-stages. We next compare the size of these LPs with that of the simultaneous-move backup of Escudie et al. (2025). Let $G \doteq \max\{|\Gamma_1|, |\Gamma_1'|\}$ bound the number of continuation representations, $K \doteq \max_{\Gamma_2 \in \Gamma_1 \cup \Gamma_1'} |\Gamma_2|$ the number of α-vectors in each representation, $C_1 \doteq |\mathcal{H}_1||\mathcal{A}_1|$, and $C_2 \doteq |\mathcal{A}_2||\mathcal{Z}_2|$. One sequential backup solves at most G LP$_1$ instances and one reduced LP$_2$.

Theorem 4.4 (Proof in App. F)**.** *The total number of nonzeros in the constraint matrices of all LPs solved in one sequential backup is $\boldsymbol{O}(GK|\mathcal{H}_2|C_1 + G|\mathcal{H}_2||\mathcal{A}_2|)$. Constructing LP$_2$ additionally requires $\boldsymbol{O}(GK|\mathcal{H}_2||\mathcal{A}_2||\mathcal{Z}_2|)$ independent α-vector evaluations outside the LP.*

In the simultaneous backup of Escudie et al. (2025), the constraint matrices contain a total of $\boldsymbol{O}(GK|\mathcal{H}_2|C_1C_2)$ nonzeros. Sequentialisation replaces this multiplicative dependence on C_1C_2 by the additive dependence in Theorem 4.4. Appendix F gives the complete variable, constraint, nonzero, and coefficient-construction counts.

Value function updates. Without pruning, adding a generated envelope cannot decrease the represented value; the formal update is given in Appendix D.

4.3 Sequential PBVI and Approximation

The value representation and polynomial backups are used as the computational core of a PBVI scheme on the sequential occupancy Markov game induced by the reduction, with sub-stages $\nu = (i,t)$ and successor $\texttt{next}(\nu)$. For each ν, the algorithm maintains sampled occupancies $\mathcal{X}_\nu$ and envelopes $\Gamma_{1,\nu}$. Each iteration performs a backward sweep over the nonterminal sub-stages, from $(2, \ell-1)$ to $(1,0)$. At each sampled occupancy, the corresponding LP update—one LP$_2$ or the collection of LP$_1$ instances—uses only $\Gamma_{1,\texttt{next}(\nu)}$ to generate a new envelope for $\Gamma_{1,\nu}$. The forward sweep then samples reachable successor occupancies. Without pruning, adding the newly generated value representations cannot decrease the represented value, as shown in Appendix D. The point-based approximation introduces error at both planning sub-stages, with no discount between them because $\gamma_1 = 1$.

Theorem 4.5 (Proof in App. E)**.** *For unpruned $\texttt{PBVI}_{\texttt{seq}}$, let $\delta \doteq \max_{i,t} \delta_{i,t}$, where $\delta_{i,t}$ is the ℓ_1-covering radius of the reachable occupancy space induced by the sampled occupancy set at (i,t), and let $m \doteq \|r\|_\infty$. The exploitability of the focal policy returned at b satisfies $\epsilon \le \frac{8m\delta}{(1-\gamma)^2}\left[1 - (\ell+1)\gamma^\ell + \ell\gamma^{\ell+1}\right]$.*

The two sub-stages contribute separate approximation errors at the same game-stage weight because $\rho_{\text{seq}} = 0$ and $\gamma_1 = 1$ at the first sub-stage. The guarantee applies to unpruned $\texttt{PBVI}_{\texttt{seq}}$; the pruning heuristics used in the experiments are not covered.

Algorithm 1 The $\texttt{PBVI}_{\texttt{seq}}$ Algorithm for $\mathcal{M}_{\text{seq}}$.

```
function PBVI_seq(ℓ)
Initialise X_ν ← ∅, for all sub-stage ν.
Initialise Γ_{1,ν} ← ∅, for all sub-stage ν.
Γ_{1,(1,ℓ)} ← {{0}}.
while has not converged do
  for ν = (2, ℓ − 1), . . . , (1, 0) do
    improve(X_ν, Γ_{1,next(ν)}).
  end for
  for ν = (1, 0), . . . , (2, ℓ − 1) do
    X_{next(ν)} ← expand(X_ν).
  end for
end while
function improve(X_ν, Γ_{1,next(ν)})
for x_ν ∈ X_ν do
  sol_ν ← LP(Γ_{1,next(ν)}, x_ν)
  Γ_{2,ν} ← Update(X_ν, Γ_{1,next(ν)}, sol_ν)
  Γ_{1,ν} ← Γ_{1,ν} ∪ {Γ_{2,ν}}
end for
function expand(X_ν)
for x_ν ∈ X_ν do
  Sample randomly a set D_ν ← {d_ν}
  X_{next(ν)} ← X_{next(ν)} ∪ {τ_seq(x_ν, d_ν) | d_ν ∈ D_ν}
end for
```

5 Sequential Transition Independence

Corollary 3.4 shows that the sequential reformulation preserves the optimal value at game-stage boundaries. We now address policy extraction and show that sequentialisation also preserves the transition-independent reduction of Escudie et al. (2025).

In the simultaneous TI-zs-SG, the simultaneous-move occupancy state is sufficient for value computation, while policy extraction relies on player-specific *occupancy sets*. For each player, this set contains the occupancies consistent with its past decision rules and the possible past decision rules of its opponent. The sequential construction uses the corresponding player occupancy sets, denoted $\mathbf{o}_{i,t}$ for player i at the corresponding planning sub-stage. Sequentialisation introduces the intermediate planning state $x_{2,t}$, which includes player 1's current action under $d_{1,t}$ in the central recursion. Player 2 receives no additional observation before acting, so its decision rule retains the original domain $d_{2,t} : \mathcal{H}_{2,t} \to \Delta(\mathcal{A}_2)$. Applying the TI-zs-SG construction to the sequential occupancy recursion defines the sequential TI-zs-SG $\mathcal{M}'_{\text{seq}}$; its formal construction is given in Appendix H.

Theorem 5.1 (Sequential lossless reduction)**.** *The sequential TI-zs-SG $\mathcal{M}'_{\text{seq}}$ is a lossless reduction of the original zs-POSG $\mathcal{M}$: it preserves policy values, equilibrium correspondence, and information structure. Composing its two planning sub-stages at each t recovers the simultaneous TI-zs-SG optimality equations of Escudie et al. (2025).*

Sequentialisation therefore splits the central planning recursion without changing the admissible policies or their returns. The approximation introduced by point-based planning is separate from this exact reduction and is analysed in Theorem 4.5.

6 Empirical Evaluation

We evaluate $\texttt{PBVI}_{\texttt{seq}}$ on Adversarial Tiger, Competitive Tiger, Recycling, and MABC (http://masplan.org/), benchmarks used in the zs-POSG literature (Wiggers et al., 2016; Delage et al., 2023; Escudie et al., 2025). We also include Kuhn Poker as a small extensive-form sanity check. Our main comparison is with $\texttt{PBVI}_{\texttt{sim}}$ (Escudie et al., 2025), which uses the simultaneous backup. All $\texttt{PBVI}_{\texttt{seq}}$ and $\texttt{PBVI}_{\texttt{sim}}$ comparisons for a given benchmark and horizon were run on the same hardware, with the same CPLEX configuration, memory limit, stopping criterion, sampling/pruning configuration, and seed set. We also compare with HSVI (Delage et al., 2023) and CFR+ (Tammelin, 2014). We report runtime, the value at the initial occupancy, and the exploitability of the returned policy.

Simultaneous versus sequential PBVI. We first examine whether the sequential backup reduces computation time when used within PBVI. Figure 1 and Table 1 show that the difference becomes pronounced at larger horizons. On Adversarial Tiger, $\texttt{PBVI}_{\texttt{seq}}$ takes 264 s at $\ell = 7$, compared with 485 s for $\texttt{PBVI}_{\texttt{sim}}$; at $\ell = 10$, the gap grows to 26.6× (287 s versus 7648 s). The same trend appears on MABC, where the runtime at $\ell = 7$ decreases from 3298 s to 178 s (18.5×), and on Recycling, where it decreases from 1969 s to 84 s (23.4×).

This behaviour is consistent with the complexity analysis of Theorem 4.4. Both methods have polynomial backup complexity in the representation stored by PBVI, but the sequential formulation separates the optimisation of the current decision rule from that of the continuation. As a result, the sequential formulation reduces the coupling in the constraint matrices of the optimisation problems, as quantified by Theorem 4.4. This difference matters little when the programmes are small, but becomes important on the larger instances, as reflected by the increasing runtime gap.

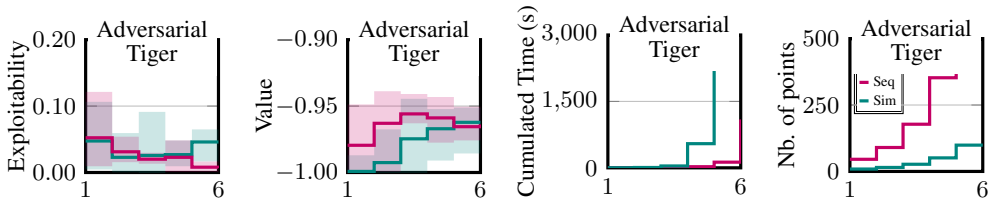

Figure 1: PBVI on Adversarial Tiger (ℓ=5): exploitability, value, cumulative runtime, and sample count over iterations, averaged over random seeds. Green: simultaneous; red: sequential.

Despite these runtime differences, the values reported by the two PBVI variants remain close. At $\ell = 10$, for example, they are -1.95 and -1.98 on Adversarial Tiger, and 0.60 and 0.62 on Recycling, for $\texttt{PBVI}_{\texttt{seq}}$ and $\texttt{PBVI}_{\texttt{sim}}$, respectively. On these benchmarks, the policies returned by $\texttt{PBVI}_{\texttt{seq}}$ also have lower exploitability at the larger horizons. On Adversarial Tiger at $\ell = 10$, exploitability decreases from 0.05 to 0.02; on MABC at $\ell = 7$, from 0.04 to 0.00; and on Recycling at $\ell = 10$, from 0.05 to 0.00.

The results on Competitive Tiger are more nuanced. At $\ell = 7$, $\texttt{PBVI}_{\texttt{seq}}$ reduces runtime from 1685 s to 450 s, while its exploitability is slightly higher (0.08 versus 0.06). The sequential method is also not always faster on the smallest instances: on Recycling at $\ell = 3$, $\texttt{PBVI}_{\texttt{sim}}$ takes 5 s, compared with 27 s for $\texttt{PBVI}_{\texttt{seq}}$. Hence, the experiments do not show uniformly better runtime or exploitability. Rather, the advantage of the sequential formulation appears on the larger instances, where the cost of the simultaneous backup becomes more significant.

Table 1: Snapshot of results (time in seconds, exploitability ε). For each setting, we report the runtime of each algorithm (in seconds), as well as the value $v(b)$ and the exploitability ε. Speedup is `Sim` time divided by `Seq` time. OOT: timeout; OOM: out of memory; '–': exploitability budget exceeded. Best values per row are highlighted in **magenta**

Game (ℓ)	Speedup	Seq			Sim			HSVI			CFR+		
		time	$v(b)$	ε	time	$v(b)$	ε	time	$v(b)$	ε	time	$v(b)$	ε
kuhn poker	0.55	0.2	**0.055**	**0.00**	0.11	**0.055**	**0.00**	NA	NA	NA	0.01	**0.055**	**0.00**
adversarial-tiger(3)	7.33	**0.15**	**-0.56**	**0.00**	1.1	**-0.56**	**0.00**	500	**-0.56**	**0.00**	1	**-0.56**	**0.00**
adversarial-tiger(4)	1.4	**10**	-0.76	**0.00**	14	-0.76	**0.00**	OOT			17	**-0.75**	0.01
adversarial-tiger(5)	1.7	**30**	-0.97	**0.00**	51	-0.97	**0.00**	OOT			181	**-0.95**	0.01
adversarial-tiger(7)	1.83	**264**	**-1.36**	**0.00**	485	-1.37	0.02	OOT			OOM		
adversarial-tiger(10)	26.6	**287**	**-1.95**	**0.02**	7648	-1.98	0.05	OOT			OOM		
adversarial-tiger(12)		**782**	**-2.38**	**0.06**	OOT			OOT			OOM		
adversarial-tiger(14)		**1840**	**-2.87**	**0.18**	OOT			OOT			OOM		
mabc(3)	20	0.85	**0.096**	**0.00**	17	**0.096**	**0.00**	70	**0.096**	**0.00**	**0.5**	**0.096**	**0.00**
mabc(4)	20.8	15	**0.11**	**0.00**	312	0.10	0.01	OOT			**4**	**0.11**	**0.00**
mabc(5)	3.1	**44**	0.11	**0.00**	136	**0.12**	0.02	OOT			51	0.12	0.01
mabc(7)	18.5	**178**	**0.13**	**0.00**	3298	**0.13**	0.04	OOT			OOM		
mabc(10)		**1377**	**0.17**	0.05	OOT			OOT			OOM		
recycling(3)	0.18	27	**0.32**	**0.00**	**5**	**0.32**	**0.00**	430	0.32	**0.00**	6	**0.32**	**0.00**
recycling(4)	0.4	36	0.35	**0.01**	**15**	**0.36**	**0.01**	OOT			80	**0.36**	0.03
recycling(5)	3.1	**72**	0.38	**0.01**	225	**0.40**	0.03	OOT			OOM		
recycling(7)	23.4	**84**	0.46	**0.01**	1969	**0.48**	0.02	OOT			OOM		
recycling(10)	14.7	**1616**	0.6	**0.00**	23798	**0.62**	0.05	OOT			OOM		
competitive-tiger(3)	0.93	48	-0.04	0.02	45	-0.04	0.02	291	**0.00**	**0.00**	**17**	**0.00**	**0.00**
competitive-tiger(4)	1.98	**60**	-0.07	0.01	119	**-0.06**	**0.00**	OOT			OOM		
competitive-tiger(5)	2.11	**152**	-0.09	0.03	322	**-0.08**	**0.02**	OOT			OOM		
competitive-tiger(7)	3.7	**450**	**-0.13**	0.08	1685	-0.15	**0.06**	OOT			OOM		
competitive-tiger(10)		**847**	**-0.23**	**0.03**	OOT			OOT			OOM		

This difference also determines which horizons can be solved within the available resources. On Adversarial Tiger, $\texttt{PBVI}_{\texttt{seq}}$ reaches $\ell = 14$ in 1840 s and returns a policy with exploitability 0.18;

on MABC, it reaches $\ell = 10$ in 1377 s with exploitability 0.05. In both cases, $\texttt{PBVI}_{\texttt{sim}}$ exceeds the computational budget. Thus, the lower cost of the sequential backup becomes particularly useful at the horizons for which the simultaneous PBVI computation no longer completes.

Other baselines. CFR+ performs well on some of the smallest instances. It solves Adversarial Tiger at $\ell = 3$ in 1 s and MABC at $\ell = 3$ in 0.5 s, with reported exploitability 0.00 in both cases. At larger horizons, however, CFR+ runs out of memory on both benchmarks at $\ell = 7$, while HSVI already exceeds the time budget at $\ell = 4$ on the reported POSG benchmarks.

7 Conclusion

We introduced a sequential planning formulation for finite-horizon two-player zero-sum POSGs. Each game stage is decomposed into two planning sub-stages, one per player, while actions remain simultaneous in the original game. We showed that sequential occupancy states are sufficient for the resulting value recursion and that the maximum-of-concave structure of the value function is preserved at both sub-stages. We also proved that the sequential reduction preserves the game value, equilibrium correspondence, and information structure of the original zs-POSG.

The sequential formulation separates the simultaneous Bellman backup into two linear programmes, which form the basis of $\texttt{PBVI}_{\texttt{seq}}$. Like the backup used by $\texttt{PBVI}_{\texttt{sim}}$, this backup is polynomial in the representation stored by PBVI. The sequential formulation, however, separates the optimisation of the current decision rule from that of the continuation, reducing the size of the optimisation problems that must be solved together. We also derived an exploitability bound for unpruned $\texttt{PBVI}_{\texttt{seq}}$ that accounts for approximation at both planning sub-stages.

The experiments show that this difference matters increasingly at larger horizons. While $\texttt{PBVI}_{\texttt{seq}}$ is not always faster on the smallest instances, it substantially reduces runtime on the more demanding ones and solves several horizons for which $\texttt{PBVI}_{\texttt{sim}}$ exceeds the available time or memory. The returned policies also have lower exploitability in several of these settings, although Competitive Tiger shows that this improvement is not uniform. Overall, the sequential formulation retains the properties of the original planning problem while making its PBVI backup cheaper to compute, allowing larger-horizon problems to be addressed within the available computational resources.

Limitations. Sequentialisation does not prevent the number of private histories or the value representation from growing with the horizon, and these remain important limits on runtime and memory. Moreover, the exploitability bound applies to unpruned $\texttt{PBVI}_{\texttt{seq}}$, whereas the experiments use pruning to keep the computation tractable. Extending the guarantee to this pruning and developing more compact value representations are natural directions for future work.

AI use statement

In this work, we used generative AI tools to assist with translation and language editing, literature search and summarisation, the development and checking of mathematical proofs, scientific figures, and software code. All mathematical claims, proofs, experimental results, and AI-assisted code were independently reviewed and verified by the authors. The authors take full responsibility for the final content of the paper and its supplementary material.

References

Richard E Bellman. *Dynamic Programming*. Dover Publications, Incorporated, 1957.

Noam Brown, Anton Bakhtin, Adam Lerer, and Qucheng Gong. Combining deep reinforcement learning and search for imperfect-information games. In *NeurIPS*, 2020.

Olivier Buffet, Jilles S. Dibangoye, Abdallah Safidine, and Vincent Thomas. ϵ-optimally solving zs-POSGs using Bellman's optimality principle. Technical report, Inria Nancy Grand-Est, 2020a.

Olivier Buffet, Jilles S. Dibangoye, Abdallah Safidine, and Vincent Thomas. Heuristic search value iteration for zero-sum stochastic games. *IEEE Transactions on Games (to appear)*, 2020b.

Rafael F Cunha, Jacopo Castellini, Johan Peralez, and Jilles S Dibangoye. On convex optimal value functions for POSGs. *arXiv preprint arXiv:2311.09459*, 2023.

Aurélien Delage, Olivier Buffet, Jilles S Dibangoye, and Abdallah Saffidine. HSVI can solve zero-sum partially observable stochastic games. *Dynamic Games and Applications*, pp. 1–55, 2023.

Jilles Dibangoye and Olivier Buffet. Learning to act in decentralized partially observable MDPs. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1233–1242, 2018.

Jilles S Dibangoye, Christopher Amato, Olivier Buffet, and François Charpillet. Optimally Solving Dec-POMDPs as Continuous-State MDPs. *Journal of Artificial Intelligence Research*, 55, 2016.

Jilles Steeve Dibangoye, Amato Christopher, and Doniec Arnaud. Scaling up decentralized mdps through heuristic search. In *Conference on Uncertainty in Artificial Intelligence*, 2012.

Jilles Steeve Dibangoye, Christopher Amato, Olivier Buffet, and François Charpillet. Optimally solving Dec-POMDPs as continuous-state MDPs. In *IJCAI*, pp. 90–96, 2013.

Erwan Escudie, Matthia Sabatelli, Olivier Buffet, and Jilles Steeve Dibangoye. ε-optimally solving two-player zero-sum POSGs. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

MK Ghosh, D McDonald, and S Sinha. Zero-sum stochastic games with partial information. *JOTA*, 121, 2004.

Eric A Hansen, Daniel S Bernstein, and Shlomo Zilberstein. Dynamic Programming for Partially Observable Stochastic Games. In *AAAI*, 2004.

Karel Horák and Branislav Bošanskỳ. Solving partially observable stochastic games with public observations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019.

Karel Horák, Branislav Bošanskỳ, and Michal Pěchouček. Heuristic search value iteration for one-sided partially observable stochastic games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

H. W. Kuhn. *Extensive games and the problem of information*. Princeton University Press, Princeton, NJ, 1953.

Matej Moravcík, Martin Schmid, Neil Burch, V. Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, K. Waugh, Michael Bradley Johanson, and Michael H. Bowling. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356:508 – 513, 2017.

Ashutosh Nayyar and Abhishek Gupta. Information structures and values in zero-sum stochastic games. In *2017 American Control Conference (ACC)*, pp. 3658–3663, 2017.

Ashutosh Nayyar, Aditya Mahajan, and Demosthenis Teneketzis. Decentralized stochastic control with partial history sharing: A common information approach. *IEEE Transactions on Automatic Control*, 58(7):1644–1658, 2013.

J. von Neumann. Zur theorie der gesellschaftsspiele. *Mathematische Annalen*, 100:295–320, 1928.

Frans A. Oliehoek, Matthijs T. J. Spaan, Jilles Steeve Dibangoye, and Christopher Amato. Heuristic search for identical payoff bayesian games. In *AAMAS*, pp. 1115–1122, 2010.

Frans Adriaan Oliehoek. Sufficient plan-time statistics for decentralized pomdps. In *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

Johan Peralez, Aurélien Delage, Olivier Buffet, and Jilles Steeve Dibangoye. Solving hierarchical information-sharing dec-POMDPs: An extensive-form game approach. In *Forty-first International Conference on Machine Learning*, 2024.

Johan Peralez, Aurélien Delage, Jacopo Castellini, Rafael F. Cunha, and Jilles S. Dibangoye. Optimally solving simultaneous-move Dec-POMDPs: The sequential central planning approach. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(22):23276–23285, Apr. 2025.

Joelle Pineau, Geoff Gordon, Sebastian Thrun, et al. Point-based value iteration: An anytime algorithm for pomdps. In *IJCAI*, volume 3, pp. 1025–1032, 2003.

Sina Sanjari, Tamer Başar, and Serdar Yüksel. Isomorphism properties of optimality and equilibrium solutions under equivalent information structure transformations: Stochastic dynamic games and teams. *SIAM Journal on Control and Optimization*, 61(5):3102–3130, 2023.

L. S. Shapley. Stochastic games*. *Proceedings of the National Academy of Sciences*, 39(10): 1095–1100, 1953.

Maurice Sion. On general minimax theorems. *Pacific Journal of Mathematics*, 8(1):171–176, 1958.

Samuel Sokota, Ryan D'Orazio, Chun Kai Ling, David J. Wu, J. Zico Kolter, and Noam Brown. Abstracting imperfect information away from two-player zero-sum games. In *International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32169–32193, 2023.

Daniel Szer, François Charpillet, and Shlomo Zilberstein. MAA*: A Heuristic Search Algorithm for Solving Decentralized POMDPs. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, 2005.

Oskari Tammelin. Solving large imperfect information games using CFR+. *CoRR*, 2014.

Auke J. Wiggers, Frans A. Oliehoek, and Diederik M. Roijers. Structure in the value function of two-player zero-sum games of incomplete information. In *ECAI*, 2016.

Appendix — Detailed Proofs

CONTENTS

1 Introduction **1**

2 Preliminaries **2**

2.1 Simultaneous-Move zs-POSGs 2

2.2 Occupancy-Based Planning 3

3 The Sequential Reformulation **3**

3.1 Sequential Occupancy States 3

3.2 Sequential Dynamics and Rewards 4

3.3 Sequential Value Recursion 4

4 Sequential Backup Geometry **5**

4.1 Sequential Value Geometry 5

4.2 Sequential Greedy Backups 5

4.3 Sequential PBVI and Approximation 6

5 Sequential Transition Independence **7**

6 Empirical evaluation **7**

7 Conclusion **9**

A Proofs for the Sequential Reformulation **12**
A.1 Sufficiency of Sequential Occupancies 12
A.2 Sequential Optimality and Value Preservation 13

B Geometry of Optimal Sequential Value Functions — Proof Theorem 4.1 **15**

C Greedy Selection Operators — Proofs of Lemmas 4.2–4.3 **16**
C.1 Proof of Lemma 4.2 16
C.2 Proof of Lemma 4.3 16

D Updating Value Function Representations **18**

E Sequential PBVI Approximation Bound **20**

F Update Complexity Reduction — Proof of Theorem 4.4 **21**

G Sequential Lossless Reduction — Proof of Theorem 5.1 **22**

H Sequential Transition-Independent Stochastic Games **23**
H.1 Formal definitions 24

I Algorithms **25**
I.1 Point Based Value Iteration 25
I.2 Pruning strategy 25

J Experimental results **26**
J.1 Benchmarks 26
J.2 Additional results 27
J.3 Implementation Details 28

A Proofs for the Sequential Reformulation

We use the objective defined in Section 3.3, with decision rules drawn from the original sets $\mathcal{D}_{i,k}$. Throughout, the initial distribution b is fixed. Discounting applies once per game stage: $\gamma_1 = 1$ and $\gamma_2 = \gamma$.

A.1 Sufficiency of Sequential Occupancies

Lemma A.1 (Sufficiency)**.** *At a fixed planning sub-stage (i, t), two admissible decision-rule prefixes inducing the same sequential occupancy induce the same remaining expected discounted return for every pair of admissible remaining decision-rule sequences. Consequently, they have the same optimal continuation value.*

Proof. First consider a prefix $\theta_{1,k}$ inducing $x_{1,k}$, and append an admissible rule $d_{1,k}$. By the definition of a decision rule,

$$\Pr(s, h_{1,k}, h_{2,k}, a_1 \mid b, \theta_{2,k}) = x_{1,k}(s, h_{1,k}, h_{2,k})\, d_{1,k}(a_1 \mid h_{1,k}) \tag{1}$$

$$= \big[\tau_{\text{seq}}(x_{1,k}, d_{1,k})\big](s, h_{1,k}, h_{2,k}, a_1). \tag{2}$$

The reward at this planning sub-stage is zero.

Next append $d_{2,k}$, so that $\theta_{1,k+1} = (\theta_{2,k}, d_{2,k})$. For $h_{i,k+1} = (h_{i,k}, a_i, z_i)$, the environment kernel gives

$$\Pr(s', h_{1,k+1}, h_{2,k+1} \mid b, \theta_{1,k+1}) \tag{3}$$

$$= \sum_s x_{2,k}(s, h_{1,k}, h_{2,k}, a_1)\, d_{2,k}(a_2 \mid h_{2,k}) \cdot p(s', z_1, z_2 \mid s, a_1, a_2) \tag{4}$$

$$= \big[\tau_{\text{seq}}(x_{2,k}, d_{2,k})\big](s', h_{1,k+1}, h_{2,k+1}). \tag{5}$$

Only s is summed out: the fixed extended histories determine the preceding histories, actions, and observations. Similarly,

$$\mathbb{E}[r(s_k, a_{1,k}, a_{2,k}) \mid b, \theta_{2,k}, d_{2,k}] \tag{6}$$

$$= \sum_{s, h_{1,k}, h_{2,k}, a_1, a_2} x_{2,k}(s, h_{1,k}, h_{2,k}, a_1) \cdot d_{2,k}(a_2 \mid h_{2,k})\, r(s, a_1, a_2) \tag{7}$$

$$= \rho_{\text{seq}}(x_{2,k}, d_{2,k}). \tag{8}$$

Now fix (i, t) and the remaining sequences $d_{1,t+i-1:}$ and $d_{2,t:}$. Equations equation 2 and equation 5, applied successively, show by induction that the occupancies generated by τ_{seq} are the conditional distributions under the extended prefixes. Equation equation 8 and linearity of expectation therefore give

$$\mathbb{E}\left[\sum_{k=t}^{\ell-1} \gamma^{k-t} r(s_k, a_{1,k}, a_{2,k}) \,\middle|\, b, \theta_{i,t}, d_{1,t+i-1:}, d_{2,t:}\right] = \sum_{k=t}^{\ell-1} \gamma^{k-t} \rho_{\text{seq}}(x_{2,k}, d_{2,k}). \tag{9}$$

The right-hand side depends on the starting prefix only through $x_{i,t}$. Thus it is identical for any two prefixes inducing that occupancy. At the same planning sub-stage, the admissible remaining sequences are also identical. Taking the maximum over $d_{1,t+i-1:}$ and the minimum over $d_{2,t:}$ proves the claim. At termination, both returns are zero. □

A.2 Sequential Optimality and Value Preservation

Lemma A.2 (Continuity and attainment)**.** *At each planning sub-stage, v^*_{seq} is continuous in the occupancy, and the extrema in its definition are attained. The one-rule objectives in Proposition 3.3 also attain their extrema.*

Proof. For every player and game stage,

$$\mathcal{D}_{i,k} = \prod_{h_{i,k} \in \mathcal{H}_{i,k}} \Delta(\mathcal{A}_i).$$

This is a finite product of compact simplices. The sets of remaining sequences are compact as well; an empty sequence forms a singleton. The return in Section 3.3 is jointly continuous in the initial occupancy and the remaining rules, since the sequential primitives involve finite sums and products. Minimising and then maximising this continuous function over the fixed compact sequence spaces yields attained extrema and a continuous value function.

The continuity can also be seen directly. For fixed remaining rules, the expected return is linear in the initial occupancy, and the absolute return from any initial state and histories is bounded by $\sum_{k=t}^{\ell-1} \gamma^{k-t} \max_{s,a_1,a_2} |r(s, a_1, a_2)|$. Since maxima and minima preserve uniform bounds on differences, occupancies at the same planning sub-stage satisfy

$$\big|v^*_{\text{seq}}(x_{i,t}) - v^*_{\text{seq}}(x'_{i,t})\big| \le \|x_{i,t} - x'_{i,t}\|_1 \left(\sum_{k=t}^{\ell-1} \gamma^{k-t}\right) \max_{s,a_1,a_2} |r(s, a_1, a_2)|. \tag{10}$$

Finally, ρ_{seq} and τ_{seq} are continuous in the current rule. Consequently, composing τ_{seq} with the continuous function v^*_{seq} gives a continuous one-rule objective on $\mathcal{D}_{i,t}$, whose extremum is attained. □

Minimax for the remaining decision rules. Fix a sequential occupancy. At the first planning sub-stage, it defines an initial chance distribution over $(s, h_{1,t}, h_{2,t})$. At the second, this distribution also includes a_1. Player 2's current information set is still determined only by $h_{2,t}$; it does not observe a_1 before acting. Subsequent transitions and observations follow p, and each player retains its own action–observation history.

In both cases, the remaining interaction is a finite extensive-form zero-sum game with perfect recall. Its behavioural strategies are precisely the admissible remaining decision-rule sequences. Kuhn's realisation-equivalence theorem allows them to be replaced by mixtures over the finitely many deterministic remaining strategies. The expected payoff is bilinear in these mixtures, so von Neumann's minimax theorem applies. Returning to behavioural strategies yields the same minimax identity, as in the occupancy-game argument of Delage et al. (2023, Theorem 3.1). Thus, for every fixed $x_{i,t}$,

$$\max_{d_{1,t+i-1:}} \min_{d_{2,t:}} \sum_{k=t}^{\ell-1} \gamma^{k-t} \rho_{\text{seq}}(x_{2,k}, d_{2,k}) = \min_{d_{2,t:}} \max_{d_{1,t+i-1:}} \sum_{k=t}^{\ell-1} \gamma^{k-t} \rho_{\text{seq}}(x_{2,k}, d_{2,k}). \tag{11}$$

The occupancies in each expression are generated from the same fixed starting occupancy under the selected rules. This argument also applies at $x_{1,t+1}$ after fixing the current rule $d_{2,t}$. It does not assume that the return is bilinear in the full behavioural decision-rule sequences.

Proof of Proposition 3.3. At $x_{1,\ell}$, the sum defining the value is empty, hence $v^*_{\text{seq}}(x_{1,\ell}) = 0$. Fix $t < \ell$.

At the first planning sub-stage, split player 1's sequence into its current rule and its remaining rules. The internal transition contributes no reward and no discount, so

$$\begin{aligned} v^*_{\text{seq}}(x_{1,t}) &= \max_{d_{1,t}} \max_{d_{1,t+1:}} \min_{d_{2,t:}} \sum_{k=t}^{\ell-1} \gamma^{k-t} \rho_{\text{seq}}(x_{2,k}, d_{2,k}) \\ &= \max_{d_{1,t}} v^*_{\text{seq}}(\tau_{\text{seq}}(x_{1,t}, d_{1,t}))\,. \end{aligned} \tag{12}$$

The second equality uses the definition of v^*_{seq} at the induced second-sub-stage occupancy. No maximum and minimum are exchanged in this step.

At the second planning sub-stage, apply equation 11 with $i = 2$, then split player 2's sequence:

$$\begin{aligned} v^*_{\text{seq}}(x_{2,t}) &= \min_{d_{2,t:}} \max_{d_{1,t+1:}} \sum_{k=t}^{\ell-1} \gamma^{k-t} \rho_{\text{seq}}(x_{2,k}, d_{2,k}) \\ &= \min_{d_{2,t}} \min_{d_{2,t+1:}} \max_{d_{1,t+1:}} \left[\rho_{\text{seq}}(x_{2,t}, d_{2,t}) + \gamma \sum_{k=t+1}^{\ell-1} \gamma^{k-t-1} \rho_{\text{seq}}(x_{2,k}, d_{2,k}) \right]. \end{aligned} \tag{13}$$

For each fixed $d_{2,t}$, the next occupancy is $x_{1,t+1} = \tau_{\text{seq}}(x_{2,t}, d_{2,t})$. The current reward is independent of the remaining rules. Applying equation 11 at this next occupancy gives

$$\min_{d_{2,t+1:}} \max_{d_{1,t+1:}} \sum_{k=t+1}^{\ell-1} \gamma^{k-t-1} \rho_{\text{seq}}(x_{2,k}, d_{2,k}) = \max_{d_{1,t+1:}} \min_{d_{2,t+1:}} \sum_{k=t+1}^{\ell-1} \gamma^{k-t-1} \rho_{\text{seq}}(x_{2,k}, d_{2,k}) \tag{14}$$

$$= v^*_{\text{seq}}(x_{1,t+1}). \tag{15}$$

Substituting equation 15 into equation 13, and using $\gamma \geq 0$, yields

$$v^*_{\text{seq}}(x_{2,t}) = \min_{d_{2,t}} \left[\rho_{\text{seq}}(x_{2,t}, d_{2,t}) + \gamma v^*_{\text{seq}}(\tau_{\text{seq}}(x_{2,t}, d_{2,t})) \right]. \tag{16}$$

All current rules range over their corresponding sets $\mathcal{D}_{i,t}$. At $t = \ell - 1$, the continuation sums are empty, so the same calculation applies. Equations equation 12 and equation 16, together with $\rho_{\text{seq}}(x_{1,t}, d_{1,t}) = 0$, $\gamma_1 = 1$, and $\gamma_2 = \gamma$, give the stated recursion. Attainment follows from Lemma A.2. □

Proof of Corollary 3.4. Fix a game-stage boundary $x_{1,t}$ and any pair of admissible remaining sequences $d_{1,t:}$ and $d_{2,t:}$. For every $k = t, \dots, \ell - 1$, substitute the internal update equation 2 into equation 5. For each fixed pair of extended histories,

$$x_{1,k+1}(s', h_{1,k+1}, h_{2,k+1}) = \sum_{s} x_{1,k}(s, h_{1,k}, h_{2,k})\, d_{1,k}(a_1 \mid h_{1,k}) d_{2,k}(a_2 \mid h_{2,k})\, p(s', z_1, z_2 \mid s, a_1, a_2). \tag{17}$$

This is the simultaneous occupancy update under the original kernel. The same substitution into equation 8 gives

$$\rho_{\text{seq}}(x_{2,k}, d_{2,k}) = \sum_{s, h_{1,k}, h_{2,k}, a_1, a_2} x_{1,k}(s, h_{1,k}, h_{2,k}) \cdot d_{1,k}(a_1 \mid h_{1,k})\, d_{2,k}(a_2 \mid h_{2,k})\, r(s, a_1, a_2). \tag{18}$$

Thus the boundary occupancies and expected rewards agree at every remaining game stage. Since there is one factor γ per game stage, the expected discounted returns agree for this fixed pair of sequences. The sets of admissible sequences are unchanged, so taking the same extrema gives

$$\begin{aligned} v^*_{\text{seq}}(x_{1,t}) &= \max_{d_{1,t:}} \min_{d_{2,t:}} \sum_{k=t}^{\ell-1} \gamma^{k-t} \rho_{\text{seq}}(x_{2,k}, d_{2,k}) \\ &= \max_{d_{1,t:}} \min_{d_{2,t:}} \mathbb{E}\left[\sum_{k=t}^{\ell-1} \gamma^{k-t} r(s_k, a_{1,k}, a_{2,k}) \,\middle|\, x_{1,t}, d_{1,t:}, d_{2,t:} \right] \\ &= v^*_{\text{sim}}(x_{1,t}). \end{aligned} \tag{19}$$

Here the expectation is taken in the original simultaneous game, and the last equality is its continuation-value definition. At $t = \ell$, both values are zero. At $t = 0$, $x_{1,0}(s, \varnothing, \varnothing) = b(s)$, hence $v^*_{\text{seq}}(b) = v^*_{\text{sim}}(b)$. □

The results above establish sufficiency for value computation and preservation of the optimal value. The correspondence with executable policies is treated in Section 5.

B Geometry of Optimal Sequential Value Functions — Proof Theorem 4.1

Proof of Theorem 4.1. Fix a sub-stage $i \in \{1, 2\}$ at stage t. For each $h_{2,t}$, define the unnormalised slice $b_{h_{2,t}}(\omega) \doteq x_i(\omega, h_{2,t})$, with $\omega \in \Omega_i$.

Fixed player-1 continuation. Fix a player 1 continuation $\pi_{1,t:}$. Each deterministic player 2 continuation tree induces a vector $\alpha^{\pi_{1,t:}, \pi_{\text{tree},2,t}} \in \mathbb{R}^{\Omega_i}$. Since the horizon and action–observation sets are finite, the set of such trees is finite; because the continuation value is linear in player 2's continuation, the minimum over mixed continuations is attained at a deterministic tree. Let Γ_2 collect these vectors. Then, for every $h_{2,t}$,

$$\min_{\pi_{\text{tree},2,t}} \left\langle b_{h_{2,t}}, \alpha^{\pi_{1,t:}, \pi_{\text{tree},2,t}} \right\rangle = \min_{\alpha \in \Gamma_2} \langle b_{h_{2,t}}, \alpha \rangle.$$

Moreover, $\pi_{2,t:}$ is indexed by player 2's histories and therefore selects, for each realised $h_{2,t}$, a re-rooted continuation tree $\pi_{\text{tree},2,t} = \pi_{2,t:}(h_{2,t})$. Hence

$$\sum_{h_{2,t}} \min_{\alpha \in \Gamma_2} \langle b_{h_{2,t}}, \alpha \rangle = \texttt{Val}_{\Gamma_2}(x_i).$$

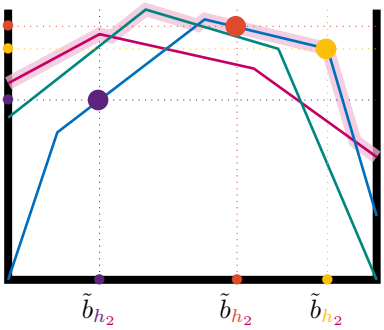

Figure 2: **Max-of-concave geometry.** For a fixed player 1 continuation, minimisation over player 2 continuations produces a piecewise-linear concave function of the local belief. Maximising over player 1 continuations forms the highlighted upper envelope. The horizontal axis shows normalized local beliefs $\tilde{b}_{h_2} \doteq b_{h_2}/\|b_{h_2}\|_1$; the occupancy representation itself uses the unnormalised slices b_{h_2}.

Optimisation over player-1 continuations. Let Γ_1 collect the finite sets Γ_2 induced by admissible player 1 continuations. Optimising over these continuations gives

$$v^*_{\text{seq}}(x_i) = \max_{\Gamma_2 \in \Gamma_1} \texttt{Val}_{\Gamma_2}(x_i),$$

as claimed. □

C Greedy Selection Operators — Proofs of Lemmas 4.2–4.3

We now turn the geometric representation into the two greedy operators used by the sequential backup.

C.1 Proof of Lemma 4.2

Proof. Fix a boundary occupancy x_1 at sub-stage $(1, t)$. By construction, $\rho_{\text{seq}}(x_1, d_1) \equiv 0$; hence any greedy decision rule of player 1 maximises

$$d_1 \longmapsto v^*_{\text{seq}}\big(\tau_{\text{seq}}(x_1, d_1)\big). \tag{20}$$

Let $x_2 \doteq \tau_{\text{seq}}(x_1, d_1)$.

At the intermediate sub-stage $(2, t)$, the optimal value admits the envelope representation

$$v^*_{\text{seq}}(x_2) = \max_{\Gamma_2 \in \Gamma_1^*} \texttt{Val}_{\Gamma_2}(x_2), \qquad \texttt{Val}_{\Gamma_2}(x_2) \doteq \sum_{h_2} \min_{\alpha \in \Gamma_2} \langle b_{h_2}, \alpha \rangle. \tag{21}$$

The maxima are attained because the remaining decision-rule spaces are compact and the finite-horizon return is continuous; see Lemma A.2.

Substituting equation 21 into equation 20 yields

$$\arg\max_{d_1 \in \mathcal{D}_1} v^*_{\text{seq}}\big(\tau_{\text{seq}}(x_1, d_1)\big) = \arg\max_{d_1 \in \mathcal{D}_1} \max_{\Gamma_2 \in \Gamma_1^*} \texttt{Val}_{\Gamma_2}\big(\tau_{\text{seq}}(x_1, d_1)\big),$$

which is the claim.

□

C.2 Proof of Lemma 4.3

Proof. Fix x_2 at sub-stage $(2, t)$.

Step 1 — Finite continuation game. The greedy Bellman step minimises $\rho_{\text{seq}}(x_2, d_2) + \gamma v^*_{\text{seq}}(\tau_{\text{seq}}(x_2, d_2))$. With finite horizon and perfect recall, Kuhn's theorem identifies behavioural

continuations with realisation-equivalent mixtures over deterministic continuations. The resulting finite zero-sum game is bilinear in these mixtures, so von Neumann's minimax theorem applies. Let $\Pi^{\mathrm{det}}_{1,t+1:}$ and $\Pi^{\mathrm{det}}_{2,t:}$ denote the finite deterministic continuation sets, and let $G_{x_2}(\pi_1, \pi_2)$ be their discounted return from x_2. The minimiser's normal-form primal is

$$\begin{aligned} \text{minimise}_{\mu,u} \quad & u \\ \text{subject to} \quad & u \geq \sum_{\pi_2 \in \Pi^{\mathrm{det}}_{2,t:}} \mu(\pi_2) G_{x_2}(\pi_1, \pi_2), \quad \forall \pi_1 \in \Pi^{\mathrm{det}}_{1,t+1:}, \\ & \mu \in \Delta(\Pi^{\mathrm{det}}_{2,t:}). \end{aligned}$$

Its feasible set is a non-empty compact simplex and its finite-horizon payoffs are bounded; the primal is therefore feasible and bounded, so finite LP strong duality applies.

Step 2 — Minimax exchange and cached continuation mixture. Dualising the finite normal-form primal exchanges the minimisation over μ with a maximisation over player 1's mixed continuation. Using the representation Γ_1' at $(1, t+1)$, this exchanged game can be written

$$\max_{\lambda \in \Delta(\Gamma_1')} \sum_{h_2} \min_{a_2} \sum_{\Gamma_2' \in \Gamma_1'} \sum_{z_2} \lambda(\Gamma_2') \min_{\alpha \in \Gamma_2'} \left\langle b_{h_2}, q^{\alpha}_{a_2, z_2} \right\rangle .$$

The same λ is shared across h_2, a_2, z_2 because it is the mixed continuation of player 1, selected before player 2's local response. By contrast, player 2 chooses a_2 before observing z_2; hence the action index must be retained until the observation contributions have been summed.

Step 3 — One-step backprojection. For each $\alpha \in \Gamma_2'$, define $q^{\alpha}_{a_2,z_2}$ as in Section 4.2. The factor $1/|\mathcal{Z}_2|$ only distributes the current reward over the observation branches, since $\sum_{z_2} r/|\mathcal{Z}_2| = r$.

$$\begin{aligned} \text{maximise}_{\lambda \in \Delta(\Gamma_1'),\, w,\, v} \quad & \textstyle\sum_{h_2} v_{h_2} \\ \text{subject to} \quad & v_{h_2} \leq \textstyle\sum_{\Gamma_2' \in \Gamma_1'} \sum_{z_2} w^{\Gamma_2'}_{h_2,a_2,z_2}, \quad \forall h_2, \forall a_2, \\ & w^{\Gamma_2'}_{h_2,a_2,z_2} \leq \lambda(\Gamma_2') \left\langle b_{h_2}, q^{\alpha}_{a_2,z_2} \right\rangle , \\ & \qquad \forall h_2, \forall a_2, \forall z_2, \forall \Gamma_2' \in \Gamma_1', \forall \alpha \in \Gamma_2'. \end{aligned} \tag{LP$_0$}$$

Step 4 — Epigraph formulation. For fixed λ, introduce $w^{\Gamma_2'}_{h_2,a_2,z_2}$ for each action–observation branch and v_{h_2} for each current private history. The inequalities

$$w^{\Gamma_2'}_{h_2,a_2,z_2} \leq \lambda(\Gamma_2') \left\langle b_{h_2}, q^{\alpha}_{a_2,z_2} \right\rangle , \qquad \forall \alpha \in \Gamma_2',$$

make each w equal at optimum to the corresponding weighted inner minimum. The inequalities

$$v_{h_2} \leq \sum_{\Gamma_2' \in \Gamma_1'} \sum_{z_2} w^{\Gamma_2'}_{h_2,a_2,z_2}, \qquad \forall a_2,$$

then make v_{h_2} equal at optimum to the minimum over current actions of the summed observation contributions. Maximising $\sum_{h_2} v_{h_2}$ over $\lambda \in \Delta(\Gamma_1')$ therefore gives exactly the finite-cache max–min above. Thus LP$_0$ in equation LP$_0$ is an exact extended formulation of the finite-cache max–min above. The next step eliminates its auxiliary branch variables to obtain LP$_2$.

Step 5 — Exact elimination of the branch variables. The branch variables w in LP$_0$ can be eliminated exactly. For fixed $\lambda \in \Delta(\Gamma_1')$, the constraints indexed by $\alpha \in \Gamma_2'$ imply

$$w_{\Gamma_2', h_2, a_2, z_2} = \min_{\alpha \in \Gamma_2'} \lambda(\Gamma_2') \left\langle b_{h_2}, q^{\alpha}_{a_2,z_2} \right\rangle$$

at optimum. Since $\lambda(\Gamma_2') \geq 0$, the common factor commutes with the inner minimum:

$$\min_{\alpha \in \Gamma_2'} \lambda(\Gamma_2') \left\langle b_{h_2}, q^{\alpha}_{a_2,z_2} \right\rangle = \lambda(\Gamma_2') \min_{\alpha \in \Gamma_2'} \left\langle b_{h_2}, q^{\alpha}_{a_2,z_2} \right\rangle .$$

The equality also holds when $\lambda(\Gamma_2') = 0$. Hence the minimising continuation vector is independent of λ. Define the branch value

$$m_{\Gamma_2',h_2,a_2,z_2} \doteq \min_{\alpha\in\Gamma_2'} \langle b_{h_2}, q^{\alpha}_{a_2,z_2} \rangle$$

and its observation-summed value

$$M_{\Gamma_2',h_2,a_2} \doteq \sum_{z_2} m_{\Gamma_2',h_2,a_2,z_2}.$$

Substituting these quantities into LP_0 eliminates all w-variables and all constraints indexed by α, yielding the equivalent reduced programme

$$\begin{aligned} &\text{maximise}_{\lambda\in\Delta(\Gamma_1'),\, v} && \textstyle\sum_{h_2} v_{h_2} \\ &\text{subject to} && v_{h_2} \le \textstyle\sum_{\Gamma_2'\in\Gamma_1'} \lambda(\Gamma_2') M_{\Gamma_2',h_2,a_2}, \qquad \forall h_2, \forall a_2. \end{aligned} \tag{LP_2}$$

Indeed, for every fixed λ, LP_2 assigns

$$v_{h_2} = \min_{a_2} \sum_{\Gamma_2'\in\Gamma_1'} \lambda(\Gamma_2') M_{\Gamma_2',h_2,a_2},$$

and therefore both formulations maximise exactly the same function

$$F(\lambda) = \sum_{h_2} \min_{a_2} \sum_{\Gamma_2'\in\Gamma_1'} \lambda(\Gamma_2') M_{\Gamma_2',h_2,a_2}.$$

Consequently, LP_0 and LP_2 have the same optimal value and the same set of optimal continuation mixtures λ. Hence eliminating w preserves both the optimal value and the set of optimal continuation mixtures λ. □

D Updating Value Function Representations

The LP solutions must be converted back into objects of the geometric representation. The next two propositions give the corresponding witness updates at the two sub-stages.

Proposition D.1 (Sub-stage 1: witness-vector update)**.** *Fix an occupancy x_1 at sub-stage 1 and a cached envelope $\Gamma_2 \in \Gamma_1$, where $\Gamma_2 \subset \mathbb{R}^{\mathcal{S}\times\mathcal{H}_1\times\mathcal{A}_1}$ indexes vectors by (s, h_1, a_1) at the intermediate sub-stage. For any decision rule d_1, define the induced intermediate slices*

$$b^{d_1}_{h_2}(s, h_1, a_1) \doteq x_1(s, h_1, h_2)\, d_1(a_1 \mid h_1).$$

For each h_2, choose an arbitrary minimiser (ties broken arbitrarily)

$$\alpha^{h_2} \in \arg\min_{\alpha\in\Gamma_2} \sum_{s,h_1,a_1} b^{d_1}_{h_2}(s, h_1, a_1)\, \alpha(s, h_1, a_1),$$

and define the generated witness vector $\beta_{d_1,\Gamma_2} \in \mathbb{R}^{\mathcal{S}\times\mathcal{H}_1\times\mathcal{H}_2}$ by

$$\beta_{d_1,\Gamma_2}(s, h_1, h_2) \doteq \sum_{a_1} d_1(a_1 \mid h_1)\, \alpha^{h_2}(s, h_1, a_1).$$

Then, for every d_1,

$$\mathtt{Val}_{\Gamma_2}\big(\tau_{\mathrm{seq}}(x_1, d_1)\big) = \sum_{s,h_1,h_2} x_1(s, h_1, h_2)\, \beta_{d_1,\Gamma_2}(s, h_1, h_2) = \langle x_1, \beta_{d_1,\Gamma_2} \rangle.$$

In particular, if (d_1^*, Γ_2^*) is the maximiser selected by Lemma 4.2, then $\beta_{d_1^*,\Gamma_2^*}$ is a valid witness vector to update the boundary envelope at x_1.

Proof. Fix d_1 and Γ_2. By the definition of $\mathtt{Val}_{\Gamma_2}$ at the sequential occupancy,

$$\mathtt{Val}_{\Gamma_2}\big(\tau_{\mathrm{seq}}(x_1, d_1)\big) = \sum_{h_2} \min_{\alpha\in\Gamma_2} \sum_{s,h_1,a_1} b^{d_1}_{h_2}(s, h_1, a_1)\, \alpha(s, h_1, a_1).$$

Since Γ_2 is finite, each inner minimum is attained; choose α^{h_2} as an arbitrary minimiser (ties broken arbitrarily). Substituting the definition of $b^{d_1}_{h_2}$ yields

$$\mathtt{Val}_{\Gamma_2}(\tau_{\text{seq}}(x_1, d_1)) = \sum_{h_2} \sum_{s,h_1,a_1} x_1(s, h_1, h_2)\, d_1(a_1 \mid h_1)\, \alpha^{h_2}(s, h_1, a_1).$$

Reordering the finite sums and using the definition of β_{d_1,Γ_2} gives

$$\sum_{s,h_1,h_2} x_1(s, h_1, h_2)\, \beta_{d_1,\Gamma_2}(s, h_1, h_2) = \langle x_1, \beta_{d_1,\Gamma_2} \rangle,$$

which proves the claim. □

Proposition D.2 (Sub-stage $(2, t)$: witness-envelope generation)**.** *Fix a reachable x_2 and a cached family Γ_1' from $(1, t+1)$. Let (λ^*, v^*) be an optimal solution of the programme in equation LP$_2$. For each $(h_2, a_2, z_2, \Gamma_2')$, choose any minimiser*

$$\alpha^{h_2,a_2,z_2,\Gamma_2'} \in \arg\min_{\alpha \in \Gamma_2'} \left\langle b_{h_2},\, q^{\alpha}_{a_2,z_2} \right\rangle,$$

and define, for each (h_2, a_2), the vector

$$\beta^{h_2,a_2} \doteq \sum_{z_2} \sum_{\Gamma_2' \in \Gamma_1'} \lambda^*(\Gamma_2')\, q^{\alpha^{h_2,a_2,z_2,\Gamma_2'}}_{a_2,z_2}.$$

Finally, define the (finite) witness envelope at x_2 as

$$\Gamma_{2,x_2} \doteq \left\{ \beta^{h_2,a_2} \;:\; h_2 \in \mathcal{H}_{2,t},\; a_2 \in \mathcal{A}_2 \right\} \subset \mathbb{R}^{\Omega_2}.$$

Then, for all h_2 and a_2,

$$\left\langle b_{h_2},\, \beta^{h_2,a_2} \right\rangle = \sum_{\Gamma_2' \in \Gamma_1'} \lambda^*(\Gamma_2') \sum_{z_2} \min_{\alpha \in \Gamma_2'} \left\langle b_{h_2},\, q^{\alpha}_{a_2,z_2} \right\rangle,$$

and therefore Γ_{2,x_2} is a valid cached envelope for evaluating $\mathtt{Val}_{\Gamma_{2,x_2}}(x_2)$, as used at predecessor boundary sub-stages (via Equation (LP$_1$)).

Proof. Because each Γ_2' is finite, the arg min is non-empty and the chosen $\alpha^{h_2,a_2,z_2,\Gamma_2'}$ attains the minimum. Substituting these minimisers into the definition of β^{h_2,a_2} gives the equality by linearity in λ^* and summation over z_2. □

Corollary D.3 (Monotone cache augmentation)**.** *Fix a sub-stage $i \in \{1, 2\}$ and let Γ_1 be the current cached family. Define the represented value function*

$$\widehat{v}_{\text{seq}}(x_i) \doteq \max_{\Gamma_2 \in \Gamma_1} \mathtt{Val}_{\Gamma_2}(x_i).$$

Let Γ_{2,x_i} be a witness envelope generated at x_i (by Proposition D.1 when $i = 1$, and by Proposition D.2 when $i = 2$), and define the augmented family $\Gamma_1^+ \doteq \Gamma_1 \cup \{\Gamma_{2,x_i}\}$, with the associated value $\widehat{v}^+_{\text{seq}}$ defined as above with Γ_1^+. Then, for every occupancy x_i,

$$\widehat{v}^+_{\text{seq}}(x_i) \;\geq\; \widehat{v}_{\text{seq}}(x_i),$$

and in particular

$$\widehat{v}^+_{\text{seq}}(x_i) = \max\left\{ \widehat{v}_{\text{seq}}(x_i),\; \mathtt{Val}_{\Gamma_{2,x_i}}(x_i) \right\}.$$

Moreover, if $\mathtt{Val}_{\Gamma_{2,x_i}}(x_i) > \widehat{v}_{\text{seq}}(x_i)$, then $\widehat{v}^+_{\text{seq}}(x_i) > \widehat{v}_{\text{seq}}(x_i)$.

Proof. By definition, $\Gamma_1 \subseteq \Gamma_1^+$, hence

$$\widehat{v}^+_{\text{seq}}(x_i) = \max_{\Gamma_2 \in \Gamma_1^+} \mathtt{Val}_{\Gamma_2}(x_i) \;\geq\; \max_{\Gamma_2 \in \Gamma_1} \mathtt{Val}_{\Gamma_2}(x_i) = \widehat{v}_{\text{seq}}(x_i).$$

The displayed identity follows because Γ_1^+ adds exactly one candidate envelope, and strict improvement holds whenever the added candidate attains a strictly larger value at x_i. □

E SEQUENTIAL PBVI APPROXIMATION BOUND

We follow the proof of Escudie et al. (2025, Theorem 7), inserting the two sequential sub-stages explicitly and using the horizon convention of this paper. Our ℓ game stages, indexed by $t = 0, \dots, \ell - 1$, correspond to planning horizon $\ell + 1$ in their notation. Let $X_{i,t} \subseteq \mathcal{X}_{i,t}$ denote the unpruned sample at sub-stage (i, t), where $\mathcal{X}_{i,t}$ is the corresponding reachable occupancy space, and define

$$\delta_{i,t} \doteq \sup_{x \in \mathcal{X}_{i,t}} \min_{\hat{x} \in X_{i,t}} \|x - \hat{x}\|_1.$$

Let $m \doteq \|r\|_\infty$ and $M_t \doteq m \sum_{k=t}^{\ell-1} \gamma^{k-t}$. For any fixed pair of remaining policies, the return from either sub-stage of game stage t is linear in its starting occupancy and its coefficient vector has sup norm at most M_t. Indeed, sub-stage $(1, t)$ has zero reward and no discount, so it has the same remaining-return bound as $(2, t)$.

Step 1 — Lipschitz error at one sampled occupancy. For any fixed continuation π and occupancies $x, \hat{x}$ at the same sub-stage,

$$|v^\pi(x) - v^\pi(\hat{x})| \le M_t \|x - \hat{x}\|_1.$$

Step 2 — From value error to exploitability. Let $\hat{x}$ be a nearest sampled occupancy and let $\hat{\pi}_1$ be the focal continuation selected by the point-based backup at $\hat{x}$. Write $\mathrm{expl}(\hat{\pi}_1; x) \doteq v^*(x) - \min_{\pi_2} v^{\hat{\pi}_1, \pi_2}(x)$. Inserting the corresponding quantities at $\hat{x}$ gives

$$\begin{aligned}\mathrm{expl}(\hat{\pi}_1; x) = \left[v^*(x) - v^*(\hat{x})\right] + \mathrm{expl}(\hat{\pi}_1; \hat{x}) \\ + \left[\min_{\pi_2} v^{\hat{\pi}_1, \pi_2}(\hat{x}) - \min_{\pi_2} v^{\hat{\pi}_1, \pi_2}(x)\right].\end{aligned}$$

Both outer terms are bounded by $M_t \|x - \hat{x}\|_1$: the first by the Lipschitz bound for v^*, and the last because the minimum of functions sharing the same Lipschitz constant is again M_t-Lipschitz. Hence

$$\mathrm{expl}(\hat{\pi}_1; x) \le \mathrm{expl}(\hat{\pi}_1; \hat{x}) + 2M_t \delta_{i,t}.$$

Following the two-term approximation decomposition of Escudie et al. (2025, Theorem 7), the same nearest-sample loss occurs once in the value approximation and once in the best-response approximation. Applying the preceding bound to both terms gives $2M_t\delta_{i,t} + 2M_t\delta_{i,t} = 4M_t\delta_{i,t}$.

Step 3 — Two sequential sub-stages. Sequentialisation creates two such approximation sites within game stage t. At $(1, t)$, $\rho_{\mathrm{seq}} = 0$ and $\gamma_1 = 1$; therefore moving to $(2, t)$ introduces no additional discount. Both local errors consequently carry the same outer game-stage weight γ^t:

$$4\gamma^t M_t \big(\delta_{1,t} + \delta_{2,t}\big).$$

Step 4 — Summation and closed form. Discounting occurs only after the $(2, t)$ transition to $(1, t + 1)$. Summing the stage contributions and substituting $M_t = m \sum_{k=t}^{\ell-1} \gamma^{k-t}$ gives

$$\epsilon \le 4m \sum_{t=0}^{\ell-1} \gamma^t \big(\delta_{1,t} + \delta_{2,t}\big) \sum_{k=t}^{\ell-1} \gamma^{k-t}.$$

Define the common covering radius $\delta \doteq \max_{i \in \{1,2\},\, t=0,\dots,\ell-1} \delta_{i,t}$. Then $\delta_{1,t} + \delta_{2,t} \le 2\delta$, and therefore

$$\epsilon \le 8m\delta \sum_{k=0}^{\ell-1} (k+1)\gamma^k = \frac{8m\delta}{(1-\gamma)^2} \left[1 - (\ell+1)\gamma^\ell + \ell\gamma^{\ell+1}\right],$$

which proves Theorem 4.5.

The separate radii are necessary because $x_{2,t}$ additionally contains a_1. The argument assumes the sampled sets are retained so that their covering radii control the approximation. It therefore applies to unpruned Sequential PBVI; bounded envelope pruning, state pruning, and numerical action thresholding are not covered.

Update	# LPs	# variables	# constraints	# dominant LP non-zeros
LP_1	$\leq G$	$O(C_1 + \lvert\mathcal{H}_2\rvert)$	$O(\lvert\mathcal{H}_2\rvert K)$	$O(GK\lvert\mathcal{H}_2\rvert C_1)$
LP_2	1	$O(G + \lvert\mathcal{H}_2\rvert)$	$O(\lvert\mathcal{H}_2\rvert\lvert\mathcal{A}_2\rvert)$	$O(G\lvert\mathcal{H}_2\rvert\lvert\mathcal{A}_2\rvert)$
		Sequential aggregate LP non-zeros		$\mathbf{O(GK\lvert\mathcal{H}_2\rvert C_1 + G\lvert\mathcal{H}_2\rvert\lvert\mathcal{A}_2\rvert)}$
Escudie et al. (2025)	1	$O(G\lvert\mathcal{H}_2\rvert C_2)$	$O(GK\lvert\mathcal{H}_2\rvert C_2)$	$\mathbf{O(GK\lvert\mathcal{H}_2\rvert C_1 C_2)}$
Delage et al. (2023)	1	rule variables + epigraph	$O(\lvert\mathcal{A}_2\rvert^{\lvert\mathcal{H}_2\rvert})$	exponential

Table 2: Greedy-backup LP complexity in common notation. Exact elimination separates coefficient construction from optimisation: LP_2 contains only continuation-mixture variables and one epigraph variable per private history, while supporting-vector evaluations are performed before the LP and can be cached. Together with the sub-stage-1 factorisation, sequentialisation removes the simultaneous $C_1 C_2$ coupling from the dominant LP constraint matrix.

F Update Complexity Reduction — Proof of Theorem 4.4

Proof of Theorem 4.4. For compactness, let $G \doteq \max\{|\Gamma_1|, |\Gamma_1'|\}$ and $K \doteq \max_{\Gamma_2 \in \Gamma_1 \cup \Gamma_1'} |\Gamma_2|$. We distinguish the size of the LPs actually solved from the cost required to construct their coefficients.

Sub-stage $(1,t)$: LP_1. Fix $\Gamma_2 \in \Gamma_1$. LP_1 has $\boldsymbol{O}(|\mathcal{H}_1||\mathcal{A}_1| + |\mathcal{H}_2|)$ variables and $\boldsymbol{O}(|\mathcal{H}_2||\Gamma_2|)$ main constraints. Across at most G such LPs, the dominant number of non-zero LP coefficients is $\boldsymbol{O}(GK|\mathcal{H}_2||\mathcal{H}_1||\mathcal{A}_1|)$.

Sub-stage $(2,t)$: coefficient construction. Exact elimination of the branch variables gives

$$m_{\Gamma_2', h_2, a_2, z_2} \doteq \min_{\alpha \in \Gamma_2'} \left\langle b_{h_2}, q^{\alpha}_{a_2, z_2} \right\rangle, \qquad M_{\Gamma_2', h_2, a_2} \doteq \sum_{z_2} m_{\Gamma_2', h_2, a_2, z_2}.$$

Constructing these coefficients requires $\boldsymbol{O}(GK|\mathcal{H}_2||\mathcal{A}_2||\mathcal{Z}_2|)$ α-vector inner-product and minimum evaluations. They are independent across $(\Gamma_2', h_2, a_2, z_2)$ and may be parallelised. An associated minimiser

$$\alpha^{\star}_{\Gamma_2', h_2, a_2, z_2} \in \arg\min_{\alpha \in \Gamma_2'} \left\langle b_{h_2}, q^{\alpha}_{a_2, z_2} \right\rangle$$

can be cached and reused during witness construction.

Sub-stage $(2,t)$: LP_2. Once M is constructed, the reduced LP contains only $\lambda \in \Delta(\Gamma_1')$ and v_{h_2}. It therefore has $\boldsymbol{O}(G + |\mathcal{H}_2|)$ variables, $\boldsymbol{O}(|\mathcal{H}_2||\mathcal{A}_2|)$ main constraints, and $\boldsymbol{O}(G|\mathcal{H}_2||\mathcal{A}_2|)$ dominant non-zero LP coefficients. The factors $K|\mathcal{Z}_2|$ have not disappeared from the backup: exact elimination moves them from the LP constraint matrix to the independent, cacheable coefficient-construction phase.

Polynomial-time conclusion. The at-most-G LP_1 instances, the coefficient-construction step, and the single LP_2 are polynomial in the explicit representation stored. Since LP is polynomial in input bit-length and coefficient construction uses polynomially many inner products, comparisons, and sums, the complete sequential backup is polynomial in the explicit representation size. □

LP complexity comparison in common notation. Let $C_1 \doteq |\mathcal{H}_1||\mathcal{A}_1|$ and $C_2 \doteq |\mathcal{A}_2||\mathcal{Z}_2|$. Our $\mathcal{Z}_2$ denotes the complete observation branch; under Escudie et al. (2025), C_2 corresponds to $|\mathcal{A}_2||\mathcal{Z}_2||\mathcal{W}|$.

Coefficient-construction cost. Before solving LP_2, constructing its coefficients requires $\boldsymbol{O}(GK|\mathcal{H}_2||\mathcal{A}_2||\mathcal{Z}_2|)$ supporting-vector inner-product/minimum evaluations. This factor has not disappeared; exact elimination moves it outside the LP constraint matrix. These evaluations may be parallelised and the corresponding $\alpha^{\star}$ cached for witness construction.

Compared with Delage et al. (2023). The earlier OMG formulation indexes constraints by all deterministic opponent decision rules, yielding an exponential description in $|\mathcal{H}_2|$. This comparison is distinct from the simultaneous formulation of Escudie et al. (2025).

Compared with Escudie et al. (2025). Both methods are polynomial in the same explicit representation stored, but the exact sequential factorisation changes where the dominant couplings occur. In the simultaneous LP, each dominant action–observation constraint can contain up to C_1 current player 1 decision-rule variables, yielding $O(GK|\mathcal{H}_2|C_1C_2)$ dominant non-zero LP coefficients. Sequentialisation first isolates the C_1 coordinates in the at-most-G LP_1 solves. Exact elimination then reduces the following optimisation to equation LP_2, whose dominant constraint matrix contains only $O(G|\mathcal{H}_2||\mathcal{A}_2|)$ non-zero coefficients. The $K|\mathcal{Z}_2|$ factor remains in the independent coefficient-construction phase, where its results can be parallelised, cached, and reused. Thus the reduction is exact: it removes these indices from the LP solved by the optimiser without omitting their arithmetic cost from the backup complexity.

G Sequential Lossless Reduction — Proof of Theorem 5.1

Proof of Theorem 5.1. Let $\mathcal{M}'$ denote the simultaneous TI-zs-SG of Escudie et al. (2025), which is a lossless reduction of the zs-POSG $\mathcal{M}$. We distinguish throughout the central occupancy states used for valuation from the player occupancy sets used for execution.

Step 1: Stage unrolling preserves the information structure. The construction of $\mathcal{M}'_{\mathrm{seq}}$ unrolls each stage t of $\mathcal{M}'$ into two internal planning sub-stages. At $(1,t)$, player 1 selects $d_{1,t}$, the central sequential occupancy state is pushed forward deterministically, the reward is 0, and $\gamma_1 = 1$. At $(2,t)$, player 2 selects $d_{2,t}$, the original TI kernel and player-occupancy-family update of $\mathcal{M}'$ are applied, the stage reward is realised, and $\gamma_2 = \gamma$.

Define Φ as the path embedding that inserts $x_{2,t}$ after each stage-boundary sequential occupancy state $x_{1,t}$, and Ψ as the projection that composes the two sub-stages at each t. On admissible trajectories, Ψ removes the uniquely determined intermediate planner state $x_{2,t} = \tau_{\mathrm{seq}}(x_{1,t}, d_{1,t})$, while Φ reconstructs it. Hence Φ and Ψ are inverse on admissible trajectories and leave the decision-rule profile unchanged. These nodes do not enlarge either player's executable information. The decision rules retain exactly the original measurable domains $d_{i,t} : \mathcal{H}_{i,t} \to \Delta(\mathcal{A}_i)$; in particular, $x_{2,t}$ is not appended to $\mathcal{H}_{2,t}$ and is not supplied to player 2 before its action at stage t. The player occupancy sets $\mathbf{o}_{i,t}$ encode the same execution information as the occupancy-set statistics of $\mathcal{M}'$, with only the activation times interleaved. Thus the same admissible decision-rule profiles index corresponding trajectories in the simultaneous and sequential models.

Under any such profile, the law of the original state, private histories, and joint actions at every game-stage boundary is identical. Rewards are realised once, at the second sub-stage, and discounting is applied once per original game stage. Therefore corresponding profiles have the same discounted return.

Step 2: Optimality and equilibrium correspondence. For any stage-boundary sequential occupancy state $x_{1,t}$,

$$v^*_{\mathrm{seq}}(x_{1,t}) = \max_{d_{1,t}} \min_{d_{2,t}} \Big[\rho_{\mathcal{M}'}(x_{1,t}, d_{1,t}, d_{2,t}) + \gamma\, v^*_{\mathrm{seq}}\big(\tau_{\mathcal{M}'}(x_{1,t}, d_{1,t}, d_{2,t})\big) \Big],$$

because $x_{2,t} = \tau_{\mathrm{seq}}(x_{1,t}, d_{1,t})$ carries neither reward nor discount, while the $(2,t)$ update applies exactly the one-stage primitives of $\mathcal{M}'$.

For fixed $x_{1,t}$, finite-horizon behavioural continuations are realization-equivalent, by Kuhn's theorem, to mixtures over finitely many deterministic continuations. The induced zero-sum return is bilinear in these mixtures on compact simplices; von Neumann's minimax theorem therefore identifies the nested max–min above with the simultaneous saddle value. Hence the displayed recursion is precisely the simultaneous TI-zs-SG optimality equation of Escudie et al. (2025). Together with the profile correspondence and return preservation of Step 1, this also preserves best responses and equilibria.

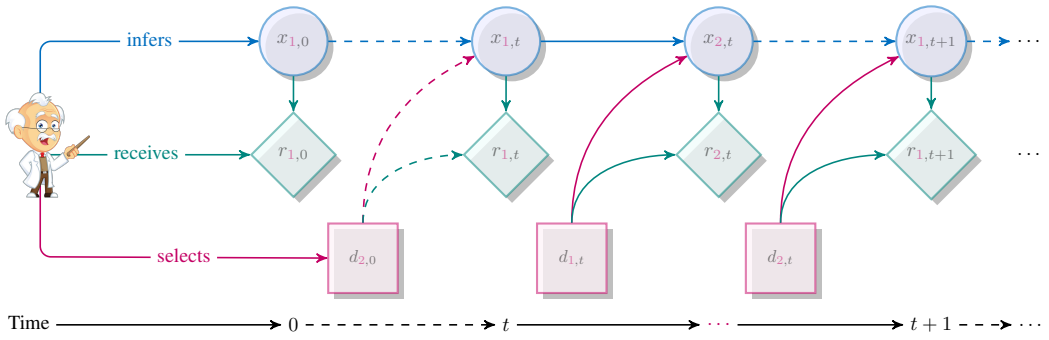

Figure 3: The influence diagram for the sequential occupancy game from the sequential central planner.

Step 3: Composition with the original zs-POSG. Since $\mathcal{M}'$ is a lossless reduction of $\mathcal{M}$, Step 1 gives information-structure and return preservation between $\mathcal{M}'$ and $\mathcal{M}'_{\text{seq}}$, and Step 2 gives value and equilibrium correspondence. Composing these correspondences proves that $\mathcal{M}'_{\text{seq}}$ is a lossless reduction of $\mathcal{M}$.

□

H Sequential Transition-Independent Stochastic Games

Roadmap. The construction keeps two levels separate throughout: sequential occupancy states $x_{i,t} \in \mathcal{X}_{\text{seq}}$ are the central statistics for valuation, whereas player occupancy sets $\mathbf{o}_{i,t} \in \mathbf{O}_i$ collect sequential occupancy states compatible with the same focal information structure and are used by the focal planning process for executable-policy extraction. We first define their timing and local updates, then the focal planning process, and finally the sequential TI-zs-SG and its compatibility operator.

This appendix formalises the *sequential* transition-independent reduction used throughout the paper. It follows the simultaneous TI-zs-SG construction of Escudie et al. (2025) and refines only the *timing* of the central recursion: each original stage t is split into two sub-stages $(1,t)$ and $(2,t)$, so that player 1 commits first and player 2 completes the stage. We present the reduction through two distinct statistics: (i) a value-sufficient *sequential occupancy state* for the central value recursion, and (ii) player-specific *occupancy families* supporting equilibrium-consistent extraction. Escudie et al. (2025) call the corresponding set-valued objects *player occupancy sets*. Here, their sequential counterparts are called *player occupancy sets* only to distinguish them from the central sequential occupancy states.

Timing. Sub-stages are indexed by (i,t) with $i \in \{1,2\}$ and $t \in \{0,\dots,\ell-1\}$. Define $\text{next}(1,t) \doteq (2,t)$ and $\text{next}(2,t) \doteq (1,t+1)$. The terminal stage boundary is $(1,\ell)$.

Sequential occupancies and occupancy families. At each sub-stage (i,t), the planner-side central state is a sequential occupancy state $x_{i,t}$, the value-sufficient statistic of the central value recursion. For executable-policy extraction, the focal planning process uses the player occupancy set $\mathbf{o}_{i,t}$; the player does not observe either this planner-side statistic or the central sequential occupancy state $x_{i,t}$ as an additional signal. The family $\mathbf{o}_{i,t}$ collects the sequential occupancy states at sub-stage (i,t) that are consistent with player i's decision-rule history up to (i,t) (with the fixed information structure), while ranging over all opponent private evolutions. Because $\mathbf{o}_{i,t}$ and $\mathbf{o}_{\text{next}(i,t)}$ live at consecutive sub-stages, the successor sequential occupancy state is recovered by a *propagate-and-match* operation: propagate the predecessor family through the occupancy recursion under the active decision rule, then match against the successor family. Thus $\mathbf{o}_{i,t}$ is a set of sequential occupancy states sharing sub-stage (i,t) and the same focal information structure; $\mathbf{O}_i$ is the corresponding occupancy-family space.

Local dynamics. At sub-stage $(1,t)$, only player 1 is active: it selects $d_{1,t}$, the central sequential occupancy state updates $x_{1,t} \mapsto x_{2,t}$, and the stage reward is 0. At sub-stage $(2,t)$, only player 2

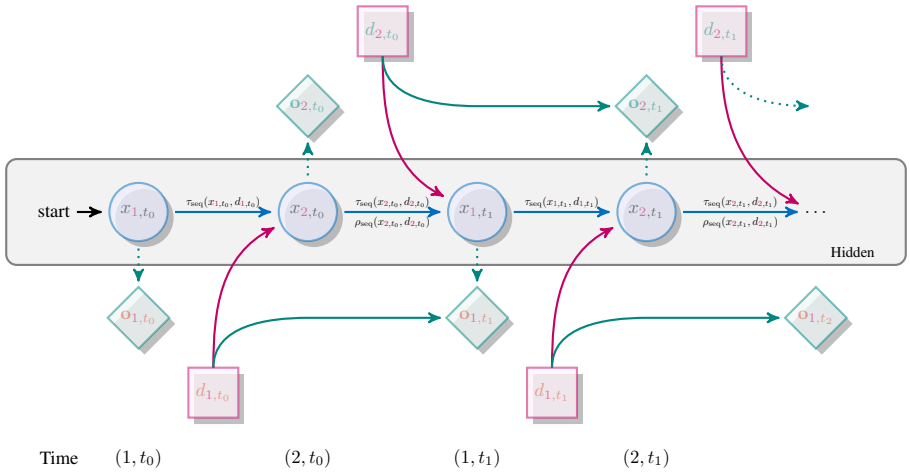

Figure 4: **Sequential Transition-Independent Stochastic Game.** The diagram illustrates the interleaved dynamics extending over stages t_0 and t_1. At each sub-stage $(1, t)$, Player 1 is active, updating their local state $\mathbf{o}_1$ and driving the transition $x_{1,t} \to x_{2,t}$. At sub-stage $(2, t)$, Player 2 acts, updating $\mathbf{o}_2$ and completing the stage transition $x_{2,t} \to x_{1,t+1}$, while generating the sequential reward ρ_{seq}.

is active: it selects $d_{2,t}$, the central sequential occupancy state updates $x_{2,t} \mapsto x_{1,t+1}$, and the original stage reward is realised. Each player-specific occupancy family evolves by appending the corresponding decision rule, exactly as in Escudie et al. (2025), but interleaved across sub-stages (Figure 4).

H.1 Formal definitions

Definition H.1 (Sequential focal planning process)**.** For each player $i \in \{1, 2\}$, the sequential focal planning process is a tuple $\mathcal{M}_{\text{seq},i} = (\mathbf{O}_i, \mathcal{F}_i, \mathcal{D}_i, \tau_{\text{seq},i}, \rho_{\text{seq},i})$, where $\mathbf{O}_i$ is the set of occupancy families $\mathbf{o}_{i,t}$, $\mathcal{F}_i \subset \mathbf{O}_i$ is the set of terminal families (at $(1, \ell)$ for $i = 1$, and after $(2, \ell - 1)$ for $i = 2$), $\mathcal{D}_i$ is the set of decision rules, $\tau_{\text{seq},i}$ maps $(\mathbf{o}_{i,t}, d_{i,t})$ to the next family at the next activation of player i, and $\rho_{\text{seq},i}$ is 0 on nonterminal families and equals $\texttt{opt}_{\neg i,\, x \in \mathbf{o}}\, g(x)$ on terminal families.

Here $\rho_{\text{seq},i}$ denotes the terminal payoff of the player-i focal planning process; it is distinct from the stage reward ρ_{seq} of the central sequential recursion in Section 3.2.

Definition H.2 (Sequential TI-zs-SG)**.** The sequential transition-independent zero-sum stochastic game is $\mathcal{M}'_{\text{seq}} = (\mathcal{X}_{\text{seq}}, \mathcal{F}_{\text{seq}}, \mathcal{M}_{\text{seq},1}, \mathcal{M}_{\text{seq},2}, \tau_{\text{seq}}, \varphi_{\text{seq}}, \rho_{\text{seq}}, \ell, \gamma)$, where $\mathcal{X}_{\text{seq}}$ and $\mathcal{F}_{\text{seq}}$ are respectively the sequential occupancy-state space and terminal set of Definition 3.2 (e.g., $x_{1,\ell}$), τ_{seq} maps $(x_{i,t}, d_{i,t})$ to $x_{\text{next}(i,t)}$, ρ_{seq} is 0 at $(1, t)$ and realises the original stage reward at $(2, t)$, and the compatibility operator is

$$\varphi_{\text{seq}}(\mathbf{o}_{i,t}, \mathbf{o}_{\text{next}(i,t)}, d_{i,t}) \doteq \left\{\tau_{\text{seq}}(x_{i,t}, d_{i,t}) \mid x_{i,t} \in \mathbf{o}_{i,t}\right\} \cap \mathbf{o}_{\text{next}(i,t)},$$

which is a singleton under the standard construction of occupancy families from decision-rule histories.[2]

[2]Occupancy families are defined as the sets of sequential occupancy states consistent with a fixed information structure and a given decision-rule history. Given $\mathbf{o}_{i,t}$ and $d_{i,t}$, propagating any $x_{i,t} \in \mathbf{o}_{i,t}$ through $\tau_{\text{seq}}(\cdot, d_{i,t})$ yields the unique successor occupancy state consistent with the successor decision-rule history; matching with $\mathbf{o}_{\text{next}(i,t)}$ therefore selects a unique element.

I ALGORITHMS

I.1 POINT BASED VALUE ITERATION

We now present a detailed description of the Point-Based Value Iteration (PBVI) algorithms for simultaneous zs-POSGs. We describe how PBVI (Pineau et al., 2003) is adapted to solve either the original model $\mathcal{M}$ (Simultaneous PBVI, *cf.* Algorithm 2) or its sequential reduction $\mathcal{M}_{\text{seq}}$ (Sequential PBVI, *cf.* Algorithm 1). In both cases, the algorithm alternates between two main phases: an *expansion* phase, which incrementally builds a representative set of reachable occupancy states, and an *improvement* phase, which updates the corresponding value function approximations over this sample.

Expansion phase – $\texttt{expand}(\mathcal{X}_\nu)$**.** Value updates are performed over a finite sample of occupancy states, rather than over the entire continuous occupancy space. This sample set is iteratively expanded by simulating likely trajectories from the initial belief state, typically by sampling decision rules according to a prescribed exploration strategy. By concentrating computation on reachable regions of the occupancy space, this procedure improves the accuracy of the value approximation. In Sequential PBVI, each original game stage corresponds to two planning sub-stages, so a horizon of ℓ game stages contains 2ℓ PBVI sub-stage updates.

Improvement phase – $\texttt{improve}(\mathcal{X}_\nu, \Gamma_{1,\texttt{next}(\nu)})$**.** Value backups are carried out backward over the sampled occupancy states, from the final stage to the initial one, in a manner analogous to backward induction. This ordering enables an efficient propagation of value information across sub-stages and between players. For each sampled occupancy, the corresponding Bellman backup optimises the active player's decision rule according to the max–min recursion. The sequential formulation replaces the simultaneous LP by at most G LP_1 instances and one reduced LP_2, with the reduction in aggregate constraint-matrix coupling characterised in Theorem 4.4.

Algorithm 2 The $\texttt{PBVI}_{\texttt{sim}}$ Algorithm for $\mathcal{M}$.

function SimPBVI(ℓ)
Initialise $\mathcal{X}_t \leftarrow \emptyset$, for all time-step t.
Initialise $\Gamma_{1,t} \leftarrow \emptyset$, for all time-steps t.
$\Gamma_{1,\ell} \leftarrow \{\{\mathbf{0}\}\}$.
while has not converged **do**
  for $t = \ell - 1, \dots, 0$ **do**
    $\texttt{improve}(\mathcal{X}_t, \Gamma_{1,t+1})$.
  end for
  for $t = 0, \dots, \ell - 1$ **do**
    $\mathcal{X}_{t+1} \leftarrow \texttt{expand}(\mathcal{X}_t)$.
  end for
end while

function $\texttt{improve}(\mathcal{X}_t, \Gamma_{1,t+1})$
for $x_t \in \mathcal{X}_t$ **do**
  $\texttt{sol}_t \leftarrow \text{LP}(\Gamma_{1,t+1}, x_t)$
  $\Gamma_{2,t} \leftarrow \texttt{Update}(\mathcal{X}_t, \Gamma_{1,t+1}, \texttt{sol}_t)$
  $\Gamma_{1,t} \leftarrow \Gamma_{1,t} \cup \{\Gamma_{2,t}\}$
end for

function expand($\mathcal{X}_t$)
for $x_t \in \mathcal{X}_t$ **do**
  Sample randomly a set $\mathcal{D} \leftarrow \{(d_{1,t}, d_{2,t})\}$
  $\mathcal{X}_{t+1} \leftarrow \mathcal{X}_{t+1} \cup \{\tau_{\text{sim}}(x_t, (d_{1,t}, d_{2,t}))|(d_{1,t}, d_{2,t}) \in \mathcal{D}\}$
end for

I.2 PRUNING STRATEGY

This section draws inspiration from (Escudie et al., 2025). For any sub-stage ν, this procedure provides the subroutines required to prune unnecessary sets of hyperplanes $\Gamma_{2,\nu}$ from the collection $\Gamma_{1,\nu}$, as well as redundant points in $\mathcal{X}_\nu$.

An envelope $\Gamma_{2,\nu} \in \Gamma_{1,\nu}$ is said to be dominated over $\mathcal{X}_\nu$ if, for every sequential occupancy state $x_\nu \in \mathcal{X}_\nu$, there exists another envelope $\Gamma'_{2,\nu} \in \Gamma_{1,\nu}$ such that $\texttt{Val}_{\Gamma_{2,\nu}}(x_\nu) < \texttt{Val}_{\Gamma'_{2,\nu}}(x_\nu)$. According to Algorithm 3, a dominated envelope $\Gamma_{2,\nu}$ has no support point in $\mathcal{X}_\nu$, and therefore satisfies $\texttt{refCount}(\Gamma_{2,\nu}) = 0$. The pruning routine consequently retains only envelopes selected at least once as maximisers over $\mathcal{X}_\nu$. Under the strict dominance definition above, every dominated envelope has zero reference count and is therefore removed.

Algorithm 3 Bounded Pruning.

```
function BoundedPruning(Γ_{1,ν}, X_ν)
for Γ_{2,ν} ∈ Γ_{1,ν} do
  refCount(Γ_{2,ν}) ← 0.
end for
for x_ν ∈ X_ν do
  Γ_{2,x_ν} ← arg max_{Γ_2 ∈ Γ_{1,ν}} Val_{Γ_2}(x_ν)
  refCount(Γ_{2,x_ν}) ← refCount(Γ_{2,x_ν}) + 1
end for
return {Γ_{2,ν} ∈ Γ_{1,ν} | refCount(Γ_{2,ν}) > 0}
```

Similarly, we remove redundant points in $\mathcal{X}_\nu$. Fix $\epsilon > 0$ and sequential occupancy state $x_\nu \in \mathcal{X}_\nu$. Let $\Gamma_{2,x_\nu} \in \Gamma_{1,\nu}$ denote an envelope within $\Gamma_{1,\nu}$ that achieves the highest pessimistic value for player 2 at x_ν, *i.e.*,

$$\mathtt{Val}_{\Gamma_{2,x_\nu}}(x_\nu) = \max_{\Gamma_2 \in \Gamma_{1,\nu}} \mathtt{Val}_{\Gamma_2}(x_\nu).$$

We say that a point $x_\nu \in \mathcal{X}_\nu$ is ϵ-redundant with respect to $(\mathcal{X}_\nu, \Gamma_{1,\nu})$ if and only if there exists another point $x'_\nu \in \mathcal{X}_\nu$ such that the gap $|\mathtt{Val}_{\Gamma_{2,x_\nu}}(x_\nu) - \mathtt{Val}_{\Gamma_{2,x'_\nu}}(x_\nu)|$ is less than or equal to ϵ. Algorithm 4 removes from $\mathcal{X}_\nu$ all ϵ-redundant points.

Algorithm 4 Redundant Sequential Occupancy State Pruning.

```
function PruneStates(X_ν, Γ_{1,ν}, ϵ)
Initialise X°_ν ← ∅
for x_ν ∈ X_ν do
  Γ_{2,x_ν} ← arg max_{Γ_2 ∈ Γ_{1,ν}} Val_{Γ_2}(x_ν)
end for
for x_ν ∈ X_ν do
  isRedundant ← false
  for x'_ν ∈ X°_ν do
    if |Val_{Γ_{2,x_ν}}(x_ν) − Val_{Γ_{2,x'_ν}}(x_ν)| ≤ ϵ then
      isRedundant ← true and break
    end if
  end for
  if ¬isRedundant then
    X°_ν ← X°_ν ∪ {x_ν}
  end if
end for
return X°_ν
```

J Experimental results

J.1 Benchmarks

We evaluate our approach on several competitive benchmark problems, adapted from standard multi-agent settings.

(Multi-Agent) Recycling. In the original cooperative version, two robots must clean a room represented as a grid by emptying garbage cans. Each robot has limited battery life and a restricted view of the environment, including limited observability of the other robot. Coordination is therefore required. We adapt the task to a zero-sum setting by altering the reward function: each robot now aims to clean more efficiently than the other.

Multi-Agent Tiger. The environment consists of two rooms—one containing a treasure and the other a tiger. Each agent stands before a door and may choose to either listen for cues or enter a room. Due to stochastic listening outcomes, agents receive noisy observations. Two competitive

variants, *Adversarial Tiger* and *Competitive Tiger*, were introduced in Wiggers et al. (2016) to study adversarial behaviour under partial observability.

Multi-Agent Broadcast Channel (MABC). This benchmark captures a communication scenario where two agents (nodes) must broadcast messages over a shared channel. To prevent collisions, only one node may broadcast at any time. While the original version is cooperative, maximising joint throughput, we consider a competitive variant by modifying the reward structure.

Kuhn Poker. Kuhn Poker is a simple two-player, zero-sum poker game with three cards. Each player is dealt one hidden card and antes one chip. Players alternate betting or passing until one folds or the bets are equal. The higher card wins the pot.

J.2 Additional results

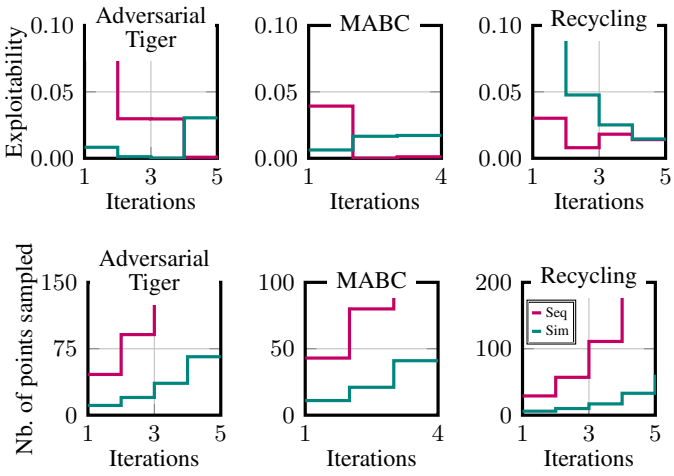

Figure 5: **Top:** Exploitability of PBVI across iterations on different benchmarks ($\ell = 5$). The green curve corresponds to the simultaneous variant, while the red curve represents the sequential variant. **Bottom:** Number of points sampled at each iteration by simultaneous and sequential variants.

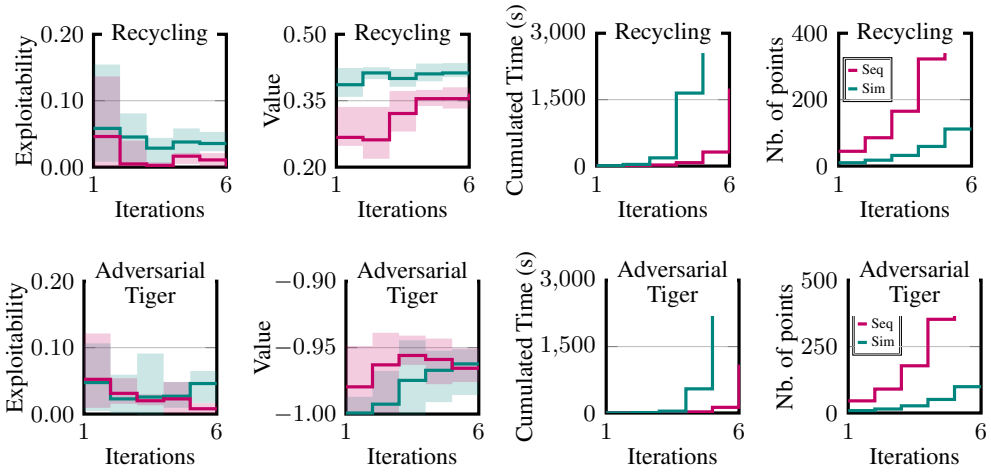

Figure 6: Exploitability, value, cumulative computation time, and number of sampled points of PBVI over iterations on the Recycling and Adversarial Tiger benchmarks (ℓ=5), evaluated across different random seeds. The green curves correspond to the simultaneous variant, while the red curves represent the sequential variant.

J.3 Implementation Details

All experiments use seed 45, and the stopping criterion is based on exploitability. The LP solver used throughout is the IBM CPLEX solver, and the timeout is benchmark-dependent. All experiments were run on CPU: for CFR+, we used an AMD Ryzen processor with up to 90GB of available memory; for HSVI/PBVI, a CPU was similarly used, with 30-70GB of available memory for smaller horizons and 90-110GB for larger horizons.

Table 3: Hyperparameters employed by $\texttt{PBVI}_{\texttt{seq}}$.

Game	Pruning threshold	Bound. Pruning	Sampling distance
kuhn poker	1e-5	1	0.0
adversarial tiger	1e-5	1	0.1
mabc	1e-5	1	0.1
recycling	1e-5	1	0.1
competitive tiger	1e-5	1	0.2

Concerning Figure 6, 5 seeds were used for each benchmark.